

Bosetti

ZX Spectrum

Tips & Tricks

EIN DATA BECKER BUCH



Bosetti

ZX Spectrum
Tips & Tricks

EIN DATA BECKER BUCH

ISBN 3-89011-075-4

Wichtiger Hinweis

Die in diesem Buch wiedergegebenen Schaltungen, Verfahren und Programme werden ohne Rücksicht auf die Patentlage mitgeteilt. Sie sind ausschließlich für Amateur- und Lehrzwecke bestimmt und dürfen nicht gewerblich genutzt werden.

Alle Schaltungen, technische Angaben und Programme in diesem Buch wurden von dem Autor mit größter Sorgfalt erarbeitet bzw. zusammengestellt und unter Einschaltung wirksamer Kontrollmaßnahmen reproduziert. Trotzdem sind Fehler nicht ganz auszuschließen. DATA BECKER sieht sich deshalb gezwungen, darauf hinzuweisen, daß weder eine Garantie noch die juristische Verantwortung oder irgendeine Haftung für Folgen, die auf fehlerhafte Angaben zurückgehen, übernommen werden kann. Für die Mitteilung eventueller Fehler ist der Autor jederzeit dankbar.

Wichtiger Hinweis:

Die in diesem Buch wiedergegebenen Schaltungen, Verfahren und Programme werden ohne Rücksicht auf die Patentlage mitgeteilt. Sie sind für Amateur- und Lehrzwecke bestimmt und dürfen nicht gewerblich genutzt werden.

Alle Schaltungen, technischen Angaben und Programme in diesem Buch wurden von dem Autor mit größter Sorgfalt erarbeitet bzw. zusammengestellt und unter Einschaltung wirksamer Kontrollmaßnahmen reproduziert. Trotzdem sind Fehler nicht ganz auszuschließen. DATA BECKER sieht sich deshalb gezwungen, darauf hinzuweisen, daß weder eine Garantie noch die juristische Verantwortung oder irgendeine Haftung für Folgen, die auf fehlerhafte Angaben zurückgehen, übernommen werden kann. Für die Mitteilung eventueller Fehler ist der Autor jederzeit dankbar.

I N H A L T

V O R W O R T

KAPITEL 1:

A L L G E M E I N E S Z U M S P E K T R U M

KAPITEL 2:

T I P S Z U R P R O G R A M M I E R U N G I N B A S I C

KAPITEL 3:

P E E K S U N D P O K E S U N D U S R S

KAPITEL 4:

N Ü T Z L I C H E R O U T I N E N U N D K U R Z E P R O G R A M M E

KAPITEL 5:

E I N S P I E L F Ü R H E L L E K Ö P F C H E N:
S Y M B O L R A T E N

KAPITEL 6:

S P E C T R U M S P E K T A K U L Ä R:
T A G U N D N A C H T A U F D E R E R D E

KAPITEL 7:

Z U S A T Z G E R Ä T E U N D
E R W E I T E R U N G E N

KAPITEL 8:

G R A F I S C H E D A R S T E L L U N G E N

KAPITEL 9:

D I E L I T F A ß S Ä U L E:
W E R B U N G M I T D E M S P E C T R U M

KAPITEL 10:

D E R S P E C T R U M I M K L E I N B E T R I E B

A N H A N G:

V O R W O R T

Hätte sich die Luftfahrt in den letzten Jahren ebenso entwickelt wie die Computerindustrie, könnten wir heute für nicht einmal 50 DM in weniger als einer halben Stunde von Frankfurt nach Tokyo fliegen.

Auf Grund dieser rasanten Entwicklung haben die Microcomputer die Welt im Sturm erobert und schon heute ist unser Leben ohne Microprozessoren kaum noch vorstellbar.

Clive Sinclair hat durch die Entwicklung von ZX-81 und ZX Spektrum einen erheblichen Anteil an der weiten Verbreitung von Computern, stellen beide Rechner doch einen preiswerten, aber lohnenden Einstieg in dies Gebiet dar.

Dieses Buch richtet sich an alle jetzigen und zukünftigen Spektrum Besitzer, die schon ein wenig vertraut mit Computern sind. Sie werden die elementaren Begriffe von Basic kennen und vielleicht schon einige Programme erstellt haben, nun aber mehr aus Ihrem Rechner herausholen wollen.

Dazu gibt dies Buch viele Anleitungen, Tips, Tricks und Routinen, welche sich zu einer Bibliothek zusammenfassen lassen und so die Erstellung eigener Programme erleichtern.

Nach einigen kurzen Programmen wird ein erstes längeres Spielprogramm entwickelt, bei dem es nicht darauf ankommt, möglichst viele Feinde zu erschiessen, sondern durch Nachdenken ein Problem zu lösen.

Im nächsten Programm sollen die grafischen Fähigkeiten des Spektrums genutzt werden. Wenn es fertig ist, zeigt der Fernsehschirm nicht nur eine Weltkarte, sondern auch den jeweiligen Sonnenstand und die Tag- und Nachtgebiete auf der Erde an.

Durch das Erscheinen der Microdrives als preiswerte, schnelle Massenspeicher wurden die Einsatzmöglichkeiten des Spektrums erheblich vergrößert. Dies umso mehr, als viele Firmen umfangreiches Zubehör (insbesondere Tastaturen) anbieten. Über Zusatzgeräte wird ausführlich berichtet.

Im weiteren werden Ideen und Programme entwickelt, welche selbst in Kleinbetrieben zu erheblichen Arbeitserleichterungen führen können, so eine Lager- und Umsatzverwaltung, eine Kundendatei und ein Grafikprogramm. Zusätzlich wird der Spektrum auch noch als Litfaßsäule eingesetzt: beliebige Worte und Texte lassen sich in verschiedenen Farben und Größen auf dem Fernsehschirm erzeugen.

Die Ausführungen des Buches sollten also für vielfältige Interessen des Lesers Hilfe und Anregung bieten und natürlich last but not least auch etwas Spaß machen.

KAPITEL 1:
A L L G E M E I N E S Z U M S P E C T R U M

Einführung

Der ZX Spectrum bietet eine preiswerte Möglichkeit, einen vielseitigen Computer zu besitzen und somit nicht nur Spaß an Spielen zu haben, sondern auch eine Menge über die 'Computerei' zu lernen. Richtig eingesetzt kann er sogar eine ganze Reihe von nützlichen Dingen erledigen.

Sollten Sie völliger Anfänger sein, so ist eine intensive Lektüre des Handbuches zum Spectrum zu empfehlen. Es ist sehr gut geschrieben und gibt nicht nur dem Einsteiger wichtige Informationen.

In diesem Buch wollen wir der Vollständigkeit halber einen kurzen Überblick über den Spectrum geben, sodaß sich zum Beispiel 'Aufsteiger' vom ZX - 81 schnell zurecht finden können.

Das Herz oder besser das Gehirn des Spectrums ist der von der Firma Zilog hergestellte Prozessor Z80 A. Dieser 8 Bit Prozessor kann einen Speicherplatz von 64 K-Byte ansprechen. 16 K-Byte dieses Speicherplatzes werden dabei vom Betriebssystem eingenommen, welches in einem ROM untergebracht ist.

Was ist ein Betriebssystem? Was ist ein ROM? Das Betriebssystem ist ein Programm, also Software, welches die Aufgabe hat, die Verbindung des Prozessors mit der Außenwelt herzustellen. Außenwelt bedeutet in diesem Fall sowohl den Benutzer des Computers als auch andere Geräte wie das Fernsehgerät oder das Kassettentonband, aber auch die Tastatur des Computers. Irgendwie muß der Spectrum ja wissen, was wir von ihm wollen.

Dies teilen wir ihm mit, indem wir bestimmte Tasten drücken. Das Betriebssystem kann erkennen, welche Bedeutung den jeweiligen Eingaben zukommt und diese verwerten. Da der Benutzer sich an bestimmte Regeln bei der Eingabe hält, kann das Programm auch entscheiden, ob gerade ein eigenes Programm geschrieben wird oder es sich um einen direkt auszuführenden Befehl handelt.

Geht es um den ersten Fall, so muß dafür gesorgt werden, daß dieses Programm erst einmal gespeichert wird und später nach Bedarf abgearbeitet werden kann. Die Sprache, mit der wir den Spectrum programmieren können, ist BASIC. Das ist eine Abkürzung für 'Beginners Allpurpose Symbolic Instruction Code'. Leider versteht der Prozessor diese Sprache nicht. Er versteht nur eine ihm eigene Maschinensprache. Daher muß ein in Basic geschriebenes Programm zunächst übersetzt werden.

Auch dieses Übersetzungsprogramm - Interpreter genannt - ist im Spectrum vorhanden und kann als Teil des Betriebssystems angesehen werden. Bei vielen Computern sind das eigentliche Betriebssystem und der Interpreter getrennte Dinge, beim Spectrum gehen sie fließend ineinander über.

Diese Programme sind natürlich sehr wichtig für den Benutzer und müssen unveränderbarer Bestandteil des Computers sein. Daher sind sie in einem Teil des Speichers, also des Gedächnisses, untergebracht, der nur zum Lesen zu benutzen ist. Daher der Name 'Read Only Memory' oder kurz ROM, was auf deutsch bedeutet 'Nur Zum Lesen'.

Wenn allerdings der gesamte Speicher des Computers nur zum Lesen da wäre, könnten wir keine eigenen Programme in den Speicher schreiben und das Gerät wäre kein richtiger Computer. Daher gibt es eine zweite Art von Speicher, der beliebig beschrieben und gelöscht werden kann. Dieser wird 'Random Access Memory' oder kurz RAM genannt.

Beim Spectrum nehmen die Programme im ROM 16 K-Byte in Anspruch. Für den RAM Speicher bleiben daher noch bis zu 48 K-Byte übrig. Davon stehen für die Basicprogrammierung etwa 40,5 KByte zur Verfügung. Diese starten erst an der Adresse 23685. Die Speicherplätze zwischen 16385 und 23685 werden für den Bildschirmspeicher und Systemvariable und ähnliche Dinge benötigt.

Da der Spectrum eine Auflösung von 256 mal 192 Punkten besitzt und jede 8 x 8 Matrix mit einem Farbattribut versehen kann werden 6144 Speicherplätze für den Bildschirmspeicher und weitere 704 für die Farbinformation benötigt.

Der gesamte Zeichenvorrat besteht beim Spectrum aus 256 Zeichen, deren Codes von 0 bis 255 durchnumeriert sind. Die ersten 128 der Zeichen entsprechen dabei allgemeinem ASCII Standart. Die Zeichen mit den Codes von 165 bis 255 repräsentieren die beim Spectrum verfügbaren Basicbefehle. Das ermöglicht, diese Befehle mit einem einzigen Tastendruck bzw. einer Kombination von Tasten eingeben zu können. Das ist nicht allein wegen der nicht gerade 'Schreibmaschinen-ähnlichen' Tastatur des Spectrums angenehm, es hat auch noch zwei weitere Vorteile. Zum einen kann man sich bei der Eingabe der Befehle nicht vertippen, zum anderen benötigen diese Befehlsworte (auch Token genannt) lediglich ein Byte im Speicher. Andere Computer benötigen oftmals für jeden Buchstaben ein Byte.

Wie erfährt das Gehirn des Spectrums, der Z80, welche Taste gedrückt wurde und was diese zu bedeuten hat? Alle Tasten sind intern miteinander verdrahtet, und zwar in Form eines Netzes, welches aus 5 mal 8 Drähten besteht. Eine Taste entspricht dann genau einer bestimmten Kombination dieser beiden Gruppen von Leitungen. Um welche Kombination es sich handelt, wird einem ganz besonderen Chip im Spectrum zugeleitet, dem ULA. Das ist eine Abkürzung für 'Uncommitted Logical Array'. Dies Chip bestimmt, um welche Taste es sich handelt. In den Systemvariablen findet der Z80 die Information, in welchem Ein-

gabemodus das System sich gerade befindet, d. h. welche Bedeutung der Taste gerade zukommt. Die einzelnen Tasten sind ja mehrfach belegt. Wenn dies ausgewertet ist, kann die Information auf den Bildschirm geleitet werden.

Es ist aber nicht nur möglich, den Spectrum in BASIC zu programmieren. Vielmehr kann man auch, sofern man dies beherrscht, in Maschinensprache arbeiten. In diesem Buch wollen wir uns aber so weit wie möglich von diesem Thema fernhalten und uns auf Basic konzentrieren. Ehe wir damit richtig beginnen, sollen noch kurz die wesentlichen Unterschiede zwischen dem guten alten ZX - 81 und dem Spectrum herausgestellt werden. Dies mag sowohl dem Aufsteiger als auch einem eventuellen Umsteiger von einem anderen System die Einarbeitungszeit für den Spectrum verkürzen.

Vom ZX - 81 zum ZX Spectrum

Wer vom Sinclair ZX - 81 zum Spectrum aufsteigt, wird nicht allzu große Probleme mit dem neuen Rechner haben, ist er doch mit der 'Sinclair - Philosophie' vertraut. Er muß sich lediglich daran gewöhnen, daß ihm ein umfangreicheres Basic zur Verfügung steht.

Im Gegensatz zum ZX - 81 ist es erlaubt, mehrere Anweisungen unter einer Zeilennummer oder im Direktmodus durch Doppelpunkt getrennt einzugeben. Da der Z80 im Spectrum nicht herangezogen wird, das Bild auf dem Schirm aufrecht zu halten wie das beim ZX - 81 ist, gibt es nicht mehr die Unterscheidung zwischen 'SLOW' und 'FAST' Modus. Der Spectrum arbeitet immer 'FAST'.

Auch der Befehl 'SCROLL' wird auf dem Spectrum im allgemeinen nicht mehr vom Basic aus benötigt. Wenn der Bildschirm voll geschrieben ist, hält er automatisch an und fragt 'SCROLL?', wonach die Ausgabe erst nach einem weiteren Tastendruck fortgesetzt wird. Möchte man dennoch die Anweisung SCROLL geben, so kann dies durch einen Aufruf einer ROM Routine (RAND USR 3582) ersetzt werden.

Durch die wesentlich höhere Grafikauflösung des Spectrums sind Unterschiede bei der Anweisung 'PLOT' zu beachten. Der Befehl 'UNPLOT' ist nicht beim Spektrum vorhanden, jedoch eine Reihe an weiteren Grafikbefehlen, die es noch nicht beim ZX - 81 gab. (Eine vollständige Liste aller Basic Anweisungen ist mit Erläuterungen im Anhang enthalten).

Beim Spectrum gibt es zwar weniger fest definierte Grafiksymbole als beim ZX - 81, jedoch stehen dem Benutzer 21 Character zur Verfügung, die er frei definieren und jederzeit aufrufen kann. Auch läßt sich beim Spectrum mit einem kleinen Trick ein völlig neuer Zeichensatz definieren, oder mehrere Zeichensätze nebeneinander benutzen. Wir werden auf diese Punkte im dritten und ins-

besondere im vierten Kapitel zurückkommen, wenn wir eine Routine entwickeln, mit deren Hilfe der Spectrum 64 Zeichen pro Zeile auf den Bildschirm ausgeben kann.

Übrigens verwendet der Spectrum im Gegensatz zum ZX - 81 den ASCII - Code.

Beim ZX Spectrum kann mit der CLEAR Anweisung auf Wunsch RAMTOP herabgesetzt werden, indem dahinter der maximale Wert dezimal eingegeben wird, der noch vom Basic erreicht werden soll. Die vom Benutzer selbst definierten Zeichen, die UDGs, liegen immer in einem vor NEW geschützten Bereich.

Die Geschwindigkeit, mit der Programme auf Kassette gespeichert werden, ist 6 mal höher als beim ZX - 81. Darüberhinaus ist das Sichern wesentlich problemloser und der SAVE Vorgang kann mit der Anweisung VERIFY überprüft werden. Ein Verlust von Daten wird somit weitestgehend ausgeschlossen. Programme können mit dem Zusatz LINE gesichert werden, was beim Laden des Programms zu einem automatischen Start des Programms an der entsprechenden Zeile führt. Beim Laden schreibt der Spectrum nun auch die Namen der Programme auf den Schirm.

Neben den acht Farben, welche beim Spectrum zur Verfügung stehen, gibt es auch etwas fürs Ohr. Mit der Anweisung 'BEEP' kann er Töne von sich geben. Beim Konzipieren dieses Befehls standen allerdings weder PINK FLOYD noch die Wiener Philharmoniker Pate. Aber immerhin, mit einigem Aufwand können ganz interessante Geräusche erzeugt werden. In einem Schach Programm kann man sogar Laute vernehmen, die Ähnlichkeit mit der menschlichen Sprache haben.

Mit dem Befehl SCREEN$ kann beim Spectrum der gesamte Bildschirminhalt auf Band gespeichert werden, auch wenn es sich um ein Gemälde in hochauflösender Grafik handelt. Das liegt daran, daß der Spectrum erlaubt, neben Basic Programmen auch Bytes zu speichern und zu laden. Der

Befehl SAVE "Name" SCREEN$ bedeutet nichts anderes als SAVE "Name" CODE 16384,6912. Vom Speicherplatz 16384 an liegt gerade 6912 Bytes lang der Displayfile. Dieser ist im Gegensatz zum ZX - 81 ein File fester Länge.

SCREEN$ kann aber auch noch anders verwendet werden. Versieht man es mit zwei Argumenten, nämlich der Zeilen- und der Spaltennummer, so erhält man als Ergebnis das Zeichen, welches sich an dieser Stelle auf dem Bildschirm befindet. Eine ähnliche Funktion erfüllt POINT (Zeile, Spalte). Hiermit kann abgefragt werden, ob ein Pixel gerade die Vorder- oder die Hintergrundfarbe hat. In diesem Fall sind mit Zeile und Spalte die Werte im Grafiksystem gemeint, also Zahlen zwischen 0 und 255 bzw. 175.

Dem Spectrum kann man Zahlen auch in binärer Form mit der Anweisung BIN eingeben. So liefert PRINT BIN 00000100 den Wert 4. Diese Möglichkeit erweist sich als besonders hilfreich, wenn man Zeichen definieren will. Man kann sich einfach auf ein 8 x 8 Raster das Zeichen aufmalen und dann an den entsprechenden Stellen eine 1 setzen. Dadurch entfällt das Umrechnen in Dezimalzahlen, was bei vielen Zeichen zu viel Arbeit führt. Ein kleines Beispiel verdeutlicht die Methode. Möchte man das 's' aus dem Schriftzug Sinclair definieren, gibt man die Zahlen

```
BIN 00000000
BIN 11111111
BIN 10000000
BIN 10000000
BIN 11111111
BIN 00000001
BIN 00000001
BIN 11111111
```

an die entsprechenden Speicherplätze ein, zum Beispiel USR "a" bis USR "a"+7. Print USR "a" zeigt dann das Zeichen in korrekter Form.

Daten lassen sich im Spectrum in DATA Anweisungen speichern und mit READ lesen. Durch RESTORE kann die Zeile, ab der gelesen werden soll, festgelegt werden.

Da das ROM des Spectrums mit seinen 16 KByte umfangreicher ist als das des ZX - 81, befinden sich die Systemvariablen auch an anderen Speicherplätzen. Die Systemvariablen und die Möglichkeiten, die sich durch PEEKen und POKEn ergeben, werden im dritten Kapitel näher besprochen.

Rein äußerlich gibt es natürlich auch einige nicht unwesentliche Unterschiede zwischen dem Spectrum und dem ZX - 81. Hier ist das bemerkenswerteste natürlich die Tastatur, die einem ZX - 81 Besitzer Jubel entlocken wird, auch wenn Benutzer von Computern anderer Marken hierfür nur ein müdes Lächeln übrig haben werden. Mit dem Spectrum + allerdings dürfte dieses Problem gänzlich ausgestanden sein.

Die Erweiterungsmöglichkeiten sind beim Spectrum, was die Firma Sinclair angeht, ganz erheblich. Mit dem Interface 1 und den Microdrives läßt sich das System so weit ausbauen, daß es den Vergleich mit teureren Geräten nicht scheuen muß. Mit den Microdrives steht ein preiswerter und relativ schneller (schneller als einige Floppy-Laufwerke) Massenspeicher zur Verfügung. Durch die eingebaute RS 232 Schnittstelle können viele Drucker angeschlossen werden, allerdings auch Verbindung mit anderen Computern, direkt oder über ein Modem, aufgenommen werden. Zusätzlich gibt es als Besonderheit das lokale Netzwerk, mit dem bis zu 64 Spectrums (und auch Sinclair QLs) direkt miteinander verbunden werden können. Über diese Dinge wird unter anderem im siebten Kapitel die Rede sein.

KAPITEL 2:
T I P S Z U R P R O G R A M M I E R U N G I N B A S I C

Allgemeines

BASIC ist die Abkürzung für Beginners Allpurpose Symbolic Instruction Code, also Symbolischer Allzweckcode für Anfänger. Ursprünglich war die Sprache also als einfacher Einstieg für Anfänger gedacht. Wie alle anderen verbreiteten Programmiersprachen auch, hat sich Basic von seinen Ursprüngen her jedoch weiterentwickelt und die Versionen, die heute bei den verschiedenen Kleincomputern Anwendung gefunden haben, zeichnen sich durch recht große Flexibilität aus.

Ohne Frage kann Basic zur Lösung vieler komplizierter Probleme sinnvoll herangezogen werden. Je umfangreicher ein Programm jedoch wird, desto größer ist die Anzahl der möglichen Fehlerquellen, und desto aufwendiger sollte daher die Vorarbeit zur eigentlichen Programmerstellung sein. Auch sollte man sich, hat man einmal das Stadium des 'Einfach Drauflosprogrammierens' überwunden, so weit wie möglich Disziplin beim Programmieren selbst auferlegen.

Ein wahlloses Herumdoktern an einem Programm führt unweigerlich zu einem Kraut- und Rübenprodukt. Wenig ist bei der Beschäftigung mit einem Computer unerfreulicher, als sich das siebzehnte Mal durch ein selbstgestricktes Programm durchzuarbeiten und die Logik nicht nachvollziehen zu können und damit auch Fehler nicht entdecken zu können.

Um sich selbst diese unnötige Arbeit und möglichst viele Fehler in den Programmen zu ersparen, empfiehlt es sich

in den meisten Fällen, das Problem zunächst mit Bleistift und Papier darzustellen. Jede Aufgabe läßt sich in eine Reihe von Teilaufgaben aufspalten, die einzeln einfacher zu lösen sind.

Berufsprogrammierer erstellen diesen Programmablauf in Form von sogenannten Flußdiagrammen. Ein solches beginnt immer mit einem Kästchen, in dem START steht und endet mit einem, in welchem STOP geschrieben ist. Dazwischen liegt der interessante Teil. Allen Teilaufgaben des Programms wird ein Kästchen zugeordnet, und diese Kästchen werden durch Pfeile miteinander verbunden. Dabei zeigen die Pfeile alle möglichen Programmabläufe an, in den Kästchen steht die jeweilige Aufgabe eines Programmteils.

Wie gesagt, für umfangreichere Programme ist die Anfertigung solcher Flußdiagramme oder zumindest einer ausführlichen Liste der Probleme unbedingt zu empfehlen. In Kapitel 4 wird ein Beispiel für ein Flußdiagramm am Beispiel des Programms Mastermind einmal durchgespielt. Ist diese Vorarbeit getan, so kann der Computer eingeschaltet werden.

Als nützlich erweist es sich, zunächst einmal eine ganze Reihe von REM Statements zu programmieren. Hier sollte man wirklich nicht geizen und nicht nur die verschiedenen Programmabschnitte kennzeichnen, sondern auch die Bedeutung der benutzten Variablen erklären. Sollte später einmal der Speicherplatz knapp werden, lassen sich die REMs wieder herauslöschen. Bis es soweit ist, leisten sie aber sehr viel Hilfe.

Eine schöne Eigenschaft der Programmiersprache Pascal ist das Konzept der sogenannten Prozeduren. Diese werden unter einem beliebigen Namen im Programm definiert und können dann jederzeit unter diesem Namen aufgerufen werden. In einigen neueren Versionen von Basic ist dies Konzept auch verwirklicht, jedoch noch nicht beim Spectrum. Wir können aber etwas ähnliches erreichen, wenn wir zu Beginn eines Programms zum Beispiel eingeben:

```
LET menu = 9000
LET aufgabe1 = 1500
      .
      .
      .
LET aufgaben = 6666
```

Werden die jeweiligen Programmabschnitte dann mit RETURN abgeschlossen, können wir die einzelnen Aufgaben mit den Anweisungen GOSUB aufgabe1 usw aufrufen. Dies macht das Programmieren wiederum viel übersichtlicher. Auch hat es den Vorteil, daß man bei Umstellungen im Programm oft nur noch die Zeilennummer für eine Aufgabe ändern muß und dann alle Aufrufe im Programm korrekt sind.

Generell empfiehlt es sich, den Programmen eine bestimmte Struktur zu geben, und diese soweit wie möglich immer beizubehalten. Eine Möglichkeit für menügesteuerte Programme ist, das Menü immer an der Zeile 9000 beginnen zu lassen. Dann können die verschiedenen Teilaufgaben bei den Zeilen 1000, 2000 usw beginnen. Die Zeilen 100 bis 999 lassen sich für Unterroutinen benutzen und die Zeilen 20 bis 99 zur Speicherreservierung für Variablen. Die ersten Zeilen können dann Farbe für Schrift, Paper und Border und ähnliches festlegen, aber auch die Zeilenzuordnung für verschiedene Teilaufgaben.

Hier ein Beispiel, welches diesen Aufbau verdeutlichen soll:

```
1 CLEAR 65535
2 INK 0: PAPER 6: BORDER 3: CLS
3 LET rand=100: LET falsch=150: LET ton=200
10 LET menu=9000: GOTO menu
20 DIM a(100): DIM b(32)
30 DIM g$(17,3)
    .
    .
90 GOTO menu
```

```
100 REM umrandung
 .
 .
140 RETURN
150 REM falsche Eingabe
 .
 .
190 RETURN
200 REM let it be beepen
 .
 .
290 RETURN
 .
 .
1000 REM erster Menupunkt
 .
 .
1990 GOTO menu
 .
 .
9000 CLS: PRINT " HAUPTMENU"
 .
 .
```

Um in diesem Programm den Speicherplatz für die Variablen zu reservieren, muß einmal zu Beginn der direkte Befehl

```
GOTO 20
```

eingegeben werden. Das Programm läßt sich mit LINE 1 saven und läuft automatisch nach dem Laden, ohne die Variablen zu löschen. Auch kann es bei einem eventuellen Absturz immer mit GOTO 1 wieder gestartet werden. Beherzigt man dies für alle Programme, braucht man sich diese Adresse nur einmal zu merken.

Um Programme relativ einfach verändern zu können oder Fehler zu suchen, empfiehlt es sich nicht nur, übersichtlich zu programmieren. Man kann auch auf einfache Weise

Programmlistings sehr 'benutzerfreundlich' gestalten. Dies geschieht nicht nur durch eine ausreichende Anzahl von REM Anweisungen, sondern man kann in seine Listings auch Farbe hereinbringen, Zeilen blinken lassen und Programmteile einrücken.

Auffallende REM Anweisungen können zum Beispiel so aussehen

```
100  REM*************************
```

oder

```
100                         REM
```

Kurze Unterroutinen können folgendermaßen eingegeben werden

```
100  REM  Loeschen der Zeilen 10 - 15
110     FOR i=10 TO 15
120       PRINT AT 10,i,,
130     NEXT i
140  RETURN
```

Auf Seite 87 des Spectrum Handbuchs sind Kontrollcodes für Farben aufgelistet. Diese können natürlich z. B. die Farbe eines Strings in einer PRINT Anweisung bestimmen, wenn man auf INK und PAPER verzichten möchte. Sie können aber auch im Listing an (fast) beliebiger Stelle eingesetzt werden. So lassen sich die Anfänge von Programmabschnitten leicht in roter Farbe herausheben, wenn vor die entsprechende Zeile die Tasten CAPS SHIFT und SYMBOL SHIFT gleichzeitig gedrückt werden, um in den Extended Mode zu gelangen, dann CAPS SHIFT und 2. Am Ende der Zeile muß diese Farbe dann abgestellt werden, indem die normale Farbe analog gewählt wird. Auf die Art kann sogar eine Zeile (oder auch das ganze Listing) zum Blinken gebracht werden.

Diese Codes lassen sich einfach merken. Zunächst muß in

den Extended Mode gegangen werden. Dann gilt einfaches Drücken einer Zahl von 0 bis 7: Einstellen der entsprechenden Farbe für den Hintergrund, 8 und 9 beeinflussen den Kontrast. Drückt man zusätzlich die CAPS SHIFT Taste, so gilt die Farbe für die Schrift, 8 und 9 stellen Blinken aus bzw. ein.

Wenn Sie ein Programm fertig erstellt haben und es ordnungsgemäß abläuft, verwenden Sie noch einige Zeit auf das Tüpfelchen auf dem i. Gestalten Sie das Programm so attraktiv wie möglich. Farbe und Ton (oder besser BEEP) sind nicht umsonst im Spectrum eingebaut und sollten nicht brach liegen. Selbst ein Schwarz-Weiss Fernsehgerät bringt die unterschiedlichen Farben in verschiedenen Grautönen, was immer noch zu einer übersichtlicheren Bildschirmgestaltung führen kann.

Programme mit gelber oder grüner Schrift auf schwarzem Hintergrund und Umrandung wirken nicht nur recht professionell, diese Farbkombination wirkt auf die Dauer nicht so anstrengend auf die Augen, wie schwarze Schrift auf weissem Untergrund. Verschiedene Schriftfarben können einen Bildschirm natürlich auch übersichtlicher machen, insbesondere wenn die gleiche Farbe immer für die gleiche Art von Information benutzt wird.

Es gibt wohl kaum ein umfangreiches Programm, welches nicht doch irgendwo einen kleinen Fehler hat. Allerdings ist es angenehm, wenn die Anzahl der Fehler so gering wie möglich ist. Meist treten Fehler in Extremsituationen im Programm auf, zum Beispiel, wenn eine Datei voll ist und dann eine bestimmte Operation durchgeführt werden soll. Auch wenn diese Operation im Normalfall immer funktioniert, schleichen sich gerade bei den seltenen Situationen die 'Bugs' ein.

Insbesondere, wenn Ihre Programme auch andere Leute benutzen werden, sollten Sie die Zeit des Austestens nie zu kurz ansetzen. Es empfiehlt sich auch, einen völlig unbedarften Menschen an das Programm zu setzen und ihn ohne

Anleitung Ihr Meisterwerk benutzen zu lassen. Wenn man ein Programm selber geschrieben hat, kommt man meist nicht auf die einfachsten Möglichkeiten, falsche Eingaben zu machen oder dem Programm zum totalen Absturz zu verhelfen.

Wenn der Speicherplatz nicht knapp wird, sollte man so viel wie möglich Hilfen zur Eingabe auf den Bildschirm schreiben. Oft weiß man selbst nach mehreren Wochen nicht mehr genau, was man in das Programm hineingeschrieben hat und ist für solche Hilfestellungen dankbar.

Effektiv programmieren

Die bisherigen Überlegungen zur sinnvollen Programmgestaltung brachten alle eine mehr oder weniger intensive Speicherplatzbelastung mit sich. Es gibt aber auch eine ganze Reihe von Möglichkeiten, sparsam mit dem Platz umzugehen, wenn dieses erforderlich ist. Ein kleines Beispiel war vorhin schon zu sehen. Nehmen wir an, Sie möchten eine Zeile, zum Beispiel die Zeile 10, löschen. Dies läßt sich sicher mit dem Befehl

```
PRINT AT 10,0;"
```

durchführen, d. h. durch das Schreiben eines leeren Strings mit 32 Elementen. Die Anweisung

```
PRINT AT 10,0,,
```

erfüllt allerdings den gleichen Zweck und benötigt 32 Bytes weniger Speicherplatz.

Zahlen werden im Spectrum mit 5 Bytes gespeichert, Keyworte dagegen benötigen nur 1 Byte. Auch werden Variablen mit einem Byte gespeichert. Damit läßt sich der obige Befehl noch effizienter eingeben, zum Beispiel als

```
     PRINT AT 10,VAL "0",,
oder PRINT AT 10,NOT PI,,
```

Die Größe PI/PI steht für '1', wie 'NOT PI' für '0' steht. Wenn Sie in einem langen Programm sehen, wie oft man in den Anweisungen die '0' oder '1' benutzt, kann man sich leicht ausrechnen, wieviel Speicherplatz im Notfall gewonnen werden kann, wenn man konsequent diese numerischen Variablen ersetzt.

Benutzt man eine Sprungadresse sehr häufig, zum Beispiel die Zeile, in der das Menü steht, so gewinnt man Platz, indem man diese Zeile mit einer (möglichst kurzen)

Variablen bezeichnet, z. B. 'm' und immer GO TO m anstelle GO TO 9000 schreibt.

Fairer Weise sollte man erwähnen, daß diese Methode ein klein wenig langsamer ist, als die direkte Angabe der Zahl, da der Spectrum sich jedesmal den Wert für die Variable suchen muß. In der Regel macht es aber keinen nennenswerten Unterschied.

Es ist sinnvoll, die gleichen Variablennamen so häufig wie möglich zu benutzen, da jede neudefinierte Variable Platz benötigt. Allerdings darf man sich auf die Art nicht Größen überschreiben, deren Wert später noch benötigt wird.

Müssen in einem Programm viele Zahlen gespeichert werden, deren Wert nicht 255 überschreitet, so geschieht dies am effektivsten, indem man anstelle der Zahl (die ganzzahlig sein muß) den Character, der dieser Zahl entspricht, speichert. Das bedarf lediglich eines einzigen Bytes gegenübr den fünf Bytes, die für jede Zahl benötigt werden. Über die Anweisung CODE wird dann die Zahl aus dem Character wiedergewonnen, wenn sie benötigt wird.

Wenn diese Zahlen bis zu 65536 reichen können, rentiert es sich immer noch, diese in zwei Bytes über die Character zu speichern. Ein Byte reicht für die Größe INT (Zahl/256), das zweite Byte für 'Zahl-256*INT (Zahl/256)'. Dies ist genau die Methode, mit welcher der Spectrum seine 65536 Speicherplätze anspricht. Im nächsten Kapitel werden wir es häufig mit dieser Adressierung zu tun haben.

Ob mit oder ohne Interface 1 läßt sich die Möglichkeit, außer der einfachen PRINT Anweisung auch PRINT# benutzen zu können, zu effektiver Programmierung verwenden, wann immer neben dem Bildschirm auch ein Drucker zur Ausgabe verwendet werden soll.

Das liegt daran, daß das Konzept der Sröme und Kanäle,

welches im Handbuch zum Interface 1 erklärt ist, auch schon ohne das zusätzliche ROM im Interface 1 vorhanden ist, allerdings in engerer Form. Eingabe und Ausgabe nämlich geschehen beim Spectrum prinzipiell über verschiedene Ströme, die in sogenannten 'Kanälen' an ihren Bestimmungsort gelangen. Im normalen ROM sind drei verschiedene Ströme vorgesehen, die mit 1, 2 und 3 bezeichnet werden. Es sind auch drei Kanäle vorhanden, einer zum unteren Bildschirm, über den INPUT und Mitteilungen des Spectrums erfolgen und der mit "k" bezeichnet wird. Der Zweite wird "s" genannt und ist für den oberen, im Normalfall größeren Bildschirmteil verantwortlich. Er wird mit PRINT angesprochen. Der Dritte ist für den Drucker verantwortlich.

Hat man das Interface 1, so gibt es noch weitere Kanäle, wie "m" für Microdrive, "t" und "b" zur Kommunikation über die serielle Schnittstelle und "n" für das Netzwerk. Diese lassen sich dann über 16 Ströme (von 0 bis 15) ansprechen. Die Anweisung PRINT "Hallo" ist eigentlich eine Kurzform für PRINT#2,"Hallo". Genauso ist die Anweisung LPRINT "Hallo" eine Kurzform für PRINT#3, "Hallo". Ganz ähnlich verhält es sich mit LIST und LLIST.

Diese Tatsache können wir aber in vielen Programmen leicht ausnutzen, um ohne viel Aufwand eine auf den Bildschirm geschriebene Information auch auf dem Drucker ausgeben zu können. Das folgende Beispiel soll dies verdeutlichen.

```
100  REM Ausgabe
110  LET ausgabe= 2
120  FOR i=1 TO 200
130  PRINT# ausgabe,a(i),b(i)
140  NEXT i
150  IF ausgabe=2 THEN INPUT "Drucken? J/N ";m$
160  IF m$="j" THEN LET ausgabe=3: GO TO 120
170  LET ausgabe=2
          .
          .
```

Wenn man in diesem Programm an der Schleife anlangt, so ist der Parameter 'ausgabe' auf 2 gesetzt und die Ausgabe erfolgt auf den Bildschirm. In Zeile 150 wird dann abgefragt, ob der Drucker benutzt werden soll. Wenn ja, wird 'ausgabe' kurzfristig auf 3 gesetzt, sodaß nun die Ausgabe wie erwünscht auf den Drucker erfolgt.

Diese Methode ist sicher, insbesondere bei umfangreichen PRINT Anweisungen sehr effektiv (und zeitsparend beim Eintippen). Auch lassen sich Ausgaben auf den Drucker mit diesem Trick leicht auf dem Schirm austesten, ohne jedes Mal Papier zu verschwenden.

KAPITEL 3:
P E E K S U N D P O K E S U N D U S R S

Der Speicher des Spectrums

Für einige Leser wird der Titel dieses Kapitels ein wenig seltsam erscheinen. Die 'Eingeweihten' wissen natürlich genau, worum es sich handelt. Wenn Sie es nicht wissen: keine Angst, nachdem Sie dieses Kapitel gelesen haben, wissen Sie Bescheid.

Wir wollen uns jetzt ein wenig mit dem Speicher des Spectrums auseinander setzen. Jeder Platz im Speicher hat eine Adresse. Wenn man wissen möchte, was an einem bestimmten Platz gespeichert ist, zum Beispiel dem Platz 1000, so gibt man die Anweisung

```
PRINT PEEK 1000
```

Der Spectrum antwortet dann damit, daß er eine Zahl zwischen 0 und 255 auf den Bildschirm schreibt (in diesem Fall eine 9).

Der Befehl PEEK sagt also 'Sieh nach, was in einem Speicherplatz steht'. Möchte man eine bestimmte Zahl, z.B. 6, an einer bestimmten Adresse abspeichern, so gibt man ein

```
POKE 30000,6.
```

Natürlich kann man nur in dem RAM - Bereich des Speichers herumpoken. Würde dies auch im ROM möglich sein, würde man ja zwangsläufig das Betriebssystem durcheinander bringen, und wer will das schon!

Die Adressen des Speichers sind auf die einfachst mögli-

che Art gewählt. Sie fangen bei 0 an und gehen je nach Version des Spectrums bis 32767 (16 K) bzw. 65535 (48 K). Die ersten 16 K-Byte, genau bis Platz 16383, sind vom ROM belegt. Hier läßt sich also nicht POKEen, sondern nur PEEKen. Allerdings können wir noch etwas mehr, wir können uns einige der Routinen des Betriebssystems zunutze machen, indem wir sie in unseren Programmen aufrufen. Wie das geschieht und wo einige für uns interessante Routinen gespeichert sind, wird gleich näher beschrieben.

In den Speicherplätzen von 16384 bis 22527 steht der Display File, also die Information, die auf dem Bildschirm zu sehen ist. Dies macht genau 256 x 192/8 =6144 Bytes aus. Jeder 8 x 8 Matrix auf dem Schirm, die genau einer Printposition entspricht, kann zusätzlich ein Farbattribut zugeordnet werden. Diese 32 x 24 = 768 Bytes schliessen sich an den Display File an und befinden sich daher an den Plätzen von 22528 bis 23295.

256 Bytes Drucker - Zwischenspeicher ist von 23296 bis 23551 untergebracht. An diesen Speicher schliessen sich die Systemvariablen an, die dem Spectrum bei der Organisation der für ein Programm wichtigen Daten helfen. Nicht alle Größen stehen nämlich an festen Adressen. Oftmals ist es sinnvoller, lediglich eine feste Adresse zu haben, in der die Adresse von gewissen Größen ist.

Das ist so ähnlich wie mit einem Handlungsreisenden, der viel unterwegs ist. Jedesmal, wenn er an einem neuen Ort angekommen ist, ruft er in seinem Heimatbüro an und gibt seine aktuelle Telefonnummer durch. Seine Mitarbeiter wissen oft nicht genau, wo er sich aufhält. Wenn sie ihn allerdings erreichen wollen, wissen sie genau, daß sie nur sein Büro anrufen müssen und man ihnen dann die Telefonnummer geben kann.

An die Systemvariablen, die eben die Auskunft über den Ort vieler Größen geben, schliessen sich Microdrive Maps und die Kanal - Information an. Dann endlich beginnt der eigentliche Basic Bereich, wobei zuerst das Programm an

sich gespeichert ist, dann die im Programm verwendeten Variablen.

Oberhalb des gesamten Basic zur Verfügung stehenden Bereichs stehen noch die User Defined Grafics Symbole.

Die Systemvariablen

Die Systemvariablen stehen an den Speicherplätzen 23552 bis 23733 für den Spectrum ohne Interface, zusätzlich werden mit dem Interface bei Benutzung der Microdrives, der RS 232 Schnittstelle und des Netzwerkes die Variablen an den Stellen 23734 bis 23791 benutzt. Einige dieser Variablen lassen sich nützlich PEEKen oder auch POKEn, bei anderen führt ungeschicktes Poken zu überraschenden Ergebnissen wie zum Beispiel einem Crash.

Die Variablen sind mit Namen versehen, die aber nicht Namen entsprechend denen in Basic sind. Um eine Variable zu verändern, kann also nicht beispielsweise die Anweisung LET VAR = 0 eingegeben werden. Vielmehr muß die entsprechende Zahl an die richtige Adresse gepoked werden.

Handelt es sich um eine Variable, die zwei Byte in Anspruch nimmt, so ist das erste Byte immer das weniger signifikante. Soll also die Dezimalzahl 'd' an die Zwei-Byte-Adresse 'z' geschrieben werden, lautet der Befehl

```
POKE z,d-256*INT (d/256)
POKE z+1,INT (d/256)
```

Beim Lesen des entsprechenden Wertes gibt man ein

```
PRINT PEEK z+256*PEEK (z+1)
```

und erhält als Ergebnis den Dezimalwert.

Die Systemvariablen sind im Spectrum Handbuch erklärt. Hier sollen einige von ihnen etwas näher erläutert werden und einige Effekte des POKEns beschrieben werden.

23560 LAST K an dieser Stelle wird der Code der letzten gedrückten Taste gespeichert. Wenn Sie diesen Wert im Direktmodus abfragen, erhalten Sie als Antwort immer 13.

23561	REPDEL	gibt die Zeit in Einheiten von 1/50 Sekunden an, ehe die gedrückte Taste automatisch wiederholt wird. Beim Kaltstart steht hier der Wert 35. Wenn Sie hier den Wert 1 hineinpoken, gibt es einen interessanten Effekt. Dieser wird noch verstärkt, wenn Sie an die nächste Adresse ebenfalls eine 1 poken. Hier steht nämlich
23562	REPPER	die Zeit zwischen zwei Wiederholungen, auch in 1/50 Sekunden. Normalerweise ist dieser 5. Es ist mehr oder weniger unmöglich, eine vernünftige Eingabe zu machen, wenn in 23561 und 23562 je eine 1 steht. Hier hilft in der Regel nur noch Netzstecker ziehen.
23606 - 23607	CHARS	hier steht die Startadresse des Zeichensatzes weniger 256. Der Zeichensatz beginnt mit dem Leerzeichen (CHR$ 32) und endet mit CHR$ 127. Die 256 werden abgezogen, da 32*8 gerade 256 ist. Diese Adresse kann sinnvoll gepoked werden, um einen eigenen Zeichensatz zu definieren. Im vierten Kapitel werden wir von dieser Möglichkeit Gebrauch machen, um 64 Zeichen pro Zeile darstellen zu können.
23608	RASP	Länge des Warntones. Dieser tönt zum Beispiel, wenn der Speicher voll ist und keine weitere Eingabe mehr verkraftet werden kann. Normalerweise steht hier eine 64. Es erscheint wenig sinnvoll, diesen Wert zu ändern.
23609	PIP	Länge des Keyboard - Klicks. Normalerweise steht hier eine Null. In manchen Programmen kann es sinnvoll sein, hier

eine Zahl hinein zu poken, um eine Eingabe akustisch zu bestätigen.

23613 - 23614 ERR SP	hier steht die Adresse, zu der bei einem Fehler gesprungen wird.
23617 MODE	gibt den Modus an, in dem man sich befindet. Ist im Normalfall 0
23624 BORDR	Farbe der Umrandung mal 8. Auch gespeichert sind die Attribute für die untere Hälfte des Bildschirms. Poken von Zahlen wie 132, 176, 198 usw führt zu netten Ergebnissen. Am einfachsten ist es, eine Schleife von 0 bis 255 zu durchfahren, aber darauf zu achten, daß bei jedem Wert auf die unteren beiden Zeilen geschrieben wird. Hier können nette Effekte erzielt werden.
23627 - 23628 VARS	gibt die Adresse der Variablen an und damit auch das Ende des Basicprogramms.
23635 - 23636 PROG	gibt den Beginn des Basicprogramms an. Zusammen mit VARS kann daher die Länge des Programms berechnet werden zu 256*(PEEK 23628-PEEK 23636) + PEEK 23627-PEEK 23635
23659 DF SZ	gibt die Anzahl der Zeilen in der unteren Bildschirmhälfte an. Dies ist normal 2. Poked man hierhin eine 0, so kann keine Fehlermeldung mehr ausgegeben werden. Das Drücken der Break-Taste führt zum Systemabsturz und verhindert die Möglichkeit, das Programm zu listen.
23672 - 23674 FRAMES	zählt die Frames, d. h. wird alle 20 Millisekunden um eins erhöht. Dies gibt die Möglichkeit, eine Echtzeitmessung

		durchzuführen, da die Erhöhung nur beim Laden oder Saven oder bei der Benutzung von BEEP unterbrochen wird. Die Anwendung wird in der Tagesschau-Uhr in Kapitel 4 verdeutlicht.
23675 - 23675	UDG	Adresse des ersten selbst definierbaren Grafiksymbols. Diese Adresse kann auch mit dem Befehl PRINT USR "a" gefunden werden. Ist der Platz schon mal sehr knapp und werden keine dieser Zeichen benötigt, kann durch Poken etwas Speicher gewonnen werden.
23677 23678	COORDS	Diese beiden Plätze geben die x und y Koordinaten des letzten geplotteten Punktes an. Diese zu poken kann manchmal nützlich sein, auch wenn PLOT den gleichen Effekt erfüllt.
23684	S POSN	gibt die Spaltenzahl der Printposition an. Hierbei wird nicht wie bei AT von 0 aufwärts, sondern von 33 abwärts gezählt
23685		analog wie 23684, jedoch für die Zeilennummer. Peeken dieser Größen kann oft von Nutzen sein.
23692	SCR CT	gibt die Anzahl der Scrolls an, ehe mit der Abfrage 'scroll' unterbrochen wird. Poked man hier immer wieder eine Zahl, die größer als 1 ist, hinein, kommt das Scrollen nicht zum Ende.
23693	ATTR P	spezifiziert die aktuellen Farben. Wie 23624 ist es effektvoll, hier hineinzupoken. Zum Beispiel 155, 156, 157 oder 0
23730 - 23731	RAMTOP	gibt die Adresse des letzten Bytes, welches für Basic zur Verfügung steht.

Kann auch mit CLEAR geändert werden.

23732 - 23733 P-RAMT — gibt die Adresse des letzten Byte des Speicherplatzes an. Dies kann sinnvoll gepeeked werden, um herauszufinden, ob es sich um die 16 oder die 48 KByte Version des Spectrums handelt. Dadurch kann in einem Programm, wo dies notwendig ist, eine automatische Anpassung vorgenommen werden.

23791 COPIES — gibt die Anzahl der Kopien, die bei Verwendung von Microdrives angefertigt werden. Gibt man hier eine große Zahl ein, kann die Zugriffszeit auf einen File verkürzt werden.

Einige Tips und Tricks

Im letzten Teil dieses Kapitels sollen noch eine Reihe von Tips aufgeführt und zusammengefaßt werden, die zum Teil erhebliche Hilfen für die Programmgestaltung oder aber für die Programmsicherung gegen unbefugtes Listen darstellen können.

Wie bereits erwähnt, kann der Spectrum ebenfalls in Maschinensprache programmiert werden. Solche Programme oder Programmteile werden von einem Basic Programm durch Befehle wie

```
        RAND USR n
oder    Print USR n
```

aufgerufen, wobei n für die Startadresse des Programms steht. Das Ende eines solchen Programms ist immer die Anweisung 'Return', nur eben in Maschinensprache.

Das gesamte ROM ist in Maschinensprache geschrieben. Dieses Programm beinhaltet eine Unzahl von Unterprogrammen, die alle, oder zumindest häufig, mit einer Return Anweisung abgeschlossen sind. Wir können also im Prinzip solche Routinen, wenn sie den gewünschten Effekt haben, aufrufen und sie uns zu Nutze machen.

Ist man es leid, den Netzstecker zu ziehen, wenn man einen Kaltstart durchführen möchte, was sicher auf Dauer nicht unbedingt für den Spectrum von Vorteil ist, so kann man dies durch einen einfachen Aufruf zum ROM ersetzen. Wir brauchen ja lediglich das ganze System von vorne beginnen zu lassen. Wie geht das? Ganz einfach, die Anweisung

```
        RAND USR 0
```

erfüllt den Zweck.

Wir können das Prinzip an einer weiteren ungefährlichen Stelle versuchen, nämlich dort, wo der Bildschirm gelöscht wird. Dies wird ab Byte 3435 durchgeführt. Also

RAND USR 3435

löscht den Bildschirm.

Nun, das hätten wir mit 'CLS' einfacher haben können. Aber nicht den Effekt, der als nächstes durchgeführt werden soll. Wir wollen nur die untere Hälfte des Bildschirms löschen. Dies geht mit

RAND USR 3652.

Wo wir gerade beim Teil des ROMs sind, der den Bildschirm bearbeitet, erinnern wir uns, daß der Spectrum zu gegebener Zeit die Abfrage 'scroll?' macht. Es muß also auch eine Routine geben, welche dieses scrolling durchführt. Je nach der Stelle, an der wir in die Routine einsteigen, erhalten wir unterschiedliche Effekte.

RAND USR 3280

scrollt eine Zeile hoch.

RAND USR 3582

scrollt eine Zeile mit Border hoch.

RAND USR 3652

scrollt die untere Hälfte des Bildes, und

RAND USR 3583

scrollt die untere Hälfte des Bildes mit Border.

Dagegen scrollt

```
RAND USR 3330
```

auf die Zeile 1.

Wie man die gesamte Farbe der Umrandung und der unteren 2 Zeilen des Schirms auf einmal beeinflußt und sogar zum Blinken bringt, hatten wir schon im letzten Abschnitt gesehen. Dies geschah mit der Anweisung

```
POKE 23624,n
```

mit Werten für n zwischen 0 und 255, wobei die höheren Werte die spektakuläreren Effekte erzielten. Das analoge gilt für die Farben des Bildschirms, wenn man eingibt

```
POKE 23693,n
```

Ist man bestrebt, seine Programme vor unbefugtem Listen zu schützen, so muß man sich schon eine ganze Reihe Tricks einfallen lassen. Es gibt ganze Firmen, die sich mit dem Kopierschutz von Software beschäftigen. Dabei ist natürlich der ideale Kopierschutz nicht gefunden worden. Das liegt daran, daß mit der Verbesserung des Schutzes auch die Methoden, diesen Schutz zu brechen, immer geschickter wurden.

Kaum jemand würde heute noch auf die Idee kommen, als einzigen Schutz PAPER und INK in der gleichen Farbe zu wählen, sodaß die Schrift nicht lesbar ist. Andererseits könnte das vielleicht ein guter Schutz sein, denn wer glaubt noch an so etwas.

Zumindest kann man sich ein Copyright - Zeichen in das Listing mit seinem Namen schreiben. Wenn man dies in Zeilennummer 1 macht und anschließend eingibt,

```
POKE 23756,0
```

so wird die Zeile in Nummer 0 umnumeriert und erscheint

nicht mehr im Listing.

Eine lustige Methode, die gegen Anfänger sehr brauchbar ist und diese möglicherweise an der Funktion ihres Computers zweifeln läßt, ist die Eingabe von

```
       POKE 23606,20
oder   POKE 23606,111
```

oder ähnlichen Werten, da dies die Schrift unleserlich macht. Dies geschieht, weil die Adresse für den Zeichensatz 'in die Wüste' geschickt wurde. Möchte man diesen Vorgang wieder rückgängig machen, muß an die Stelle eine 0 gepoked werden.

Natürlich kann man diese Veränderung der Adresse des Zeichensatzes auch sinnvoll nutzen, um sich einen eigenen Zeichensatz zu definieren. Dies wird im nächsten Kapitel ausführlich besprochen, wenn wir den Spectrum dazu bringen, mit 64 Zeichen pro Zeile auf den Bildschirm zu schreiben.

Eine einfache Methode, ein Ausbrechen aus einem Programm zu verhindern, besteht darin, die Zahl der Zeilen im unteren Bildschirm, die im Normalfall 2 ist, auf 0 zu setzen durch

```
POKE 23659,0.
```

Dann ist kein Platz mehr für eine Fehlermeldung, wie sie auch ein BREAK hervorruft, und es kommt zum Systemcrash.

Ganz analog funktioniert der Befehl

```
POKE 23613,0: POKE 23614,0
```

da hiermit das Programm bei einer Fehlermeldung einen Kaltstart verursacht. Der einzige Nachteil ist, daß diese Variablen nicht immer so erhalten bleiben, wie sie

gepoked werden. Der Vorgang muß also öfters wiederholt werden.

KAPITEL 4:

N Ü T Z L I C H E R O U T I N E N U N D K U R Z E P R O G R A M M E

Schrift in doppelter Größe

Nun wird es Zeit, daß wir uns mit handfesten Dingen beschäftigen. Beginnen wir mit einigen Routinen, welche die Schriftgrösse auf dem Bildschirm beeinflussen.

Als erstes wollen wir versuchen, den Inhalt eines Feldes, zum Beispiel y$, in doppelter Höhe darzustellen. Das würde Überschriften oder andere wichtige Mitteilungen in einem Programm herausheben.

Fangen wir zunächst mit einem einzelnen Zeichen an. Als erstes holen wir uns den Code des Zeichens. Dieser beginnt an der Stelle 8*CODE y$ + PEEK 23606 + 256*PEEK 23607 = 8* CODE y$ + 15360 und besteht aus 8 Zahlen. Nehmen wir nun die ersten vier Zahlen her und stecken sie jeweils zweimal hintereinander in USR "a" und die zweiten vier analog in USR "b", so brauchen wir nur noch diese beiden selbstdefinierten Zeichen untereinander an die gewünschte Stelle zu printen und sind dann fertig.

Also:

```
800 LET x=8*CODE y$ +15360
810 FOR z=0 TO 7
820 LET a=PEEK (x+z)
830 LET b=PEEK (x+z+1)
840 POKE USR "a"+2*z,a
850 POKE USR "a"+2*z+1,b
860 NEXT z
```

und zum Beispiel

```
900 PRINT AT 9,9;"A"
910 PRINT AT 10,9;"B"
```

wobei in den letzten beiden Zeilen die entsprechenden Grafikzeichen zu wählen sind.

Geben wir nun noch ein

```
700 INPUT y$
```

und 990 GOTO 700

gefolgt von RUN, so können wir die Routine austesten.

Wenn alles ordnungsgemäß abläuft, wollen wir die Routine noch derart erweitern, daß y$ mehr als nur ein Zeichen lang sein darf. Dazu bilden wir eine Schleife über die Länge von y$:

```
750 FOR i=1 TO LEN y$
```

Dann müssen wir die Zeilen 800, 900 und 910 ändern

```
800 LET x=8*CODE y$(i)+15360
900 PRINT AT 9,i;"A"
910 PRINT AT 10,i;"B"
```

und hinzufügen

```
920 NEXT i
```

Indem man die Printposition für 'Grafik A' und 'Grafik B' zum Beispiel als 'xpos,ypos' und 'xpos+1,ypos' wählt und ein 930 RETURN ranhängt, ist dies eine nützliche Unterroutine, die viele Programme ansprechender gestalten hilft.

Zusammenhängend lautet die Routine dann:

```
750 FOR i=1 TO LEN y$
```

```
800 LET x=8*CODE y(i)+15360
810 FOR z=0 TO 7
820 LET a=PEEK (x+z)
830 LET b=PEEK (x+z+1)
840 POKE USR "a"+2*z,a
850 POKE USR "b"+2*z+1,b
860 NEXT z
900 PRINT AT xpos,ypos+i-1;"A"
910 PRINT AT xpos+1,ypos+i-1;"B"
920 NEXT i
930 RETURN
```

Die Routine ist relativ langsam, was aber nicht immer als störend angesehen werden muß. Falls man es ein wenig schneller haben möchte, kann man die folgende Version wählen:

```
800 LET c=USR "a"
810 FOR i=1 TO LEN y$
820 IF y$(i)=" " THEN PRINT AT xpos+1,ypos+i-1;" "
    : GO TO 900
830 LET x=8*CODE y$(i)+15360
840 FOR z=0 TO 7
850 POKE c+2*z,PEEK (x+z)
860 POKE c+2*z+1,PEEK (x+z+1)
870 NEXT z
880 PRINT AT xpos,ypos+i-1;"A"
890 PRINT AT xpos+1,ypos+i-1;"B"
900 NEXT i
910 RETURN
```

Hier sind in den Schleifen weniger Rechnungen notwendig, sodaß diese Version fast doppelt so schnell wird. Vergleichen Sie die beiden Routinen einmal. Denken Sie aber bitte daran, daß die Größen xpos und ypos definiert werden müssen.

Möchte man noch größere Buchstaben auf ähnliche Art darstellen, kommt man schnell an die Grenzen von Basic, die Routinen dauern einfach zu lang. In diesen Fällen sollte

man dazu übergehen, in Maschinensprache zu programmieren. Im Kapitel 9 werden wir eine Routine in Maschinensprache benutzen, um Buchstaben in jeder beliebigen Größe schreiben zu können. Sie können aber auch Ihre Routinen selber in Basic schreiben, und diese dann mit Hilfe eines Compilers in Maschinensprache übersetzen. Über diese Compiler wird im Kapitel 7 geschrieben.

64 Zeichen pro Zeile ausgeben

Nachdem wir Möglichkeiten gefunden haben, Buchstaben und Zeichen größer als normal darstellen zu können, wollen wir jetzt versuchen, kleinere Buchstaben auf den Bildschirm zu bekommen. Warum dies? Nun, in manchem Programm reichen die 32 Zeichen pro Zeile nicht aus oder sind zumindest unbequem, wenn zum Beispiel einige Zahlenkolonnen oder viel Text gezeigt werden soll. Nicht umsonst gibt es für einige Computer sogenannte 80 Zeichen Karten für eine Menge Geld zu kaufen.

Das Textverarbeitungsprogramm Tasword II ist in der Lage, 64 Zeichen pro Zeile darzustellen. Über dieses Programm wird auch im Kapitel 7 geschrieben. Hier wollen wir eine sehr einfache Methode entwickeln, in eigenen Programmen auf Wunsch 64 Zeichen pro Zeile ausgeben zu können. Der Aufwand lohnt sich bestimmt, es kann doppelt so viel Information auf dem Bildschirm gezeigt werden, als im Normalmodus.

Wie funktioniert das Ganze? Nun, jedes Zeichen wird vom Spectrum in einer Matrix von 8 x 8 Punkten oder Pixels dargestellt. Man ist aber auch in der Lage, jedes Zeichen in einer Matrix von 4 x 8 Pixels zu zeigen, sodaß die Zeichen dennoch gut lesbar sind. Probieren wir dies einfach einmal aus. Dazu benutzen wir zunächst die UDGs, also die User defined Grafics Symbole.

Versuchen wir es mit dem Buchstaben 'A'. In binärer Form kann dieser Buchstabe folgendermaßen dargestellt werden:

```
BIN 00000000
BIN 00000100
BIN 00001010
BIN 00001010
BIN 00001110
BIN 00001010
BIN 00001010
BIN 00000000
```

Sicher erkennen Sie das Muster des Buchstabens. Bringen wir diesen auf den Schirm. Dazu geben wir die folgenden Anweisungen ein:

```
10   DATA BIN 00000000,BIN 00000100,BIN 00001010,BIN 0000
     1010,BIN 00001110,BIN 00001010,BIN 00001010,BIN 0000
     0000
20   FOR i=0 TO 7
30   READ x
40   POKE USR "a"+i,x
50   NEXT i
100  CLS
110  FOR i=1 TO 300
120  PRINT "A";
130  NEXT i
140  STOP
```

In Zeile 120 wird natürlich Grafik A eingegeben, also CHR$ 144. Wenn Sie dieses kleine Programm eingegeben haben und RUN drücken, sehen Sie 300 schmale A's auf dem Bildschirm. So weit so gut, das ist noch ganz einfach ein neuer Schriftsatz, wie wir ihn in Kapitel 3 besprochen haben. Wie können wir die Zeichen nun näher aneinander rücken, sodaß auch wirklich mehr Zeichen auf dem Bildschirm erscheinen?

Der nächste Schritt auf dem Weg dorthin ist eine kleine Ergänzung zum obigen Programm. Geben wir in dieses ein:

```
15   DATA BIN 00000000,BIN 00100000,BIN 01010000,BIN 0101
     0000,BIN 01110000,BIN 01010000,BIN 01010000,BIN 0000
     0000

60   FOR i=0 TO 7
70   READ x
80   POKE USR "b"+i,x
90   NEXT i
```

```
125  PRINT "B";
```

Wieder ist der gedruckte Buchstabe der Grafik Character. Nun geben Sie wieder RUN ein. Sie sehen jetzt 600 kleine A's, wobei abwechseln einmal nur die rechte, einmal nur die linke Hälfte einer 8 x 8 Matrix ausgenutzt ist. Damit sind wir zwar noch nicht am Ziel, aber doch schon ein ganzes Stück näher.

Jetzt müssen wir den Spectrum nur noch dazu bringen, beide Zeichen auf die gleiche Position zu bringen, ohne daß sie sich gegenseitig löschen. Das letztere läßt sich durch OVER 1 natürlich leicht verwirklichen. Wenn wir also nach jedem Zeichen, welches die linke Hälfte der Matrix benutzt, die Print - Position um eins zurück setzen, ist das Problem gelöst. Das Zurücksetzten geschieht einfach mit PRINT CHR$ 8. Wir müssen dann allerdings in unserem Beispiel mit dem Grafik B beginnen, da dieser die linke Hälfte der Matrix nutzt. Ändern wir also das Programm folgendermaßen um:

```
120  PRINT "B";CHR$ 8;
125  PRINT OVER 1;"A";
```

und geben wieder RUN ein. Nun, ist das nichts?

Jetzt ist der weitere Weg vorgezeichnet: wir werden uns einen vollständigen Zeichensatz definieren, bei dem nur die linke Hälfte der Matrix genutzt wird und einen für die rechte Hälfte. Diese beiden Zeichensätze werden wir dann im Speicher irgendwo oberhalb RAMTOP speichern. Wenn wir dann etwas mit 64 Zeichen pro Zeile ausgeben wollen, setzen wir den Zeiger auf den Charactersatz, der in den Systemvariablen 23606 und 23607 untergebracht ist, abwechselnd zum einen und anderen Zeichensatz während wir den Text entsprechend unserem kleinen Programm auf den Schirm bringen.

Die Zeichen, die wir darstellen wollen, umfassen diejenigen mit den Codes von 32 bis 143, also 112 Zeichen.

Dazu benötigen wir 8 * 112 Zahlen pro Zeichensatz, für beide Zeichensätze sind das 2 * 896 Zahlen. Der normale Zeichensatz wird mit der Adresse 15360 angesteuert, das bedeutet, daß im Speicherplatz 23606 eine 0, in 23607 eine 60 steht, da 0 + 256*60 gerade 15360 ergibt. Der eigentliche Zeichensatz beginnt an der Stelle 15516, die druckbaren Zeichen aber erst 256 Plätze später.

Es empfiehlt sich, die neuen Zeichensätze an Adressen beginnen zu lassen, die Vielfache von 256 sind, da dann beim Umschalten nur eine Adresse gePOKEd werden muß, nämlich 23607.

Nehmen wir im Augenblick an, wir hätten die beiden Zeichensätze bereits definiert und an die Speicherplätze von 63488 bis 64383 und 64512 bis 65408 geschrieben. Dann sichern wir diese mit dem Befehl

```
SAVE "64MC" CODE 63488,1921 bzw.
SAVE *"m";1;"64MC" CODE 63488,1921
```

Die Speicheradressen sind gerade so gewählt, daß mit den Befehlen POKE 23607,247 und POKE 23607,251 die neuen Zeichensätze aktiviert werden können. Mit POKE 23607,60 kann dann wieder auf die normalen Zeichen zurückgestellt werden. Schreiben wir uns also jetzt eine kleine Routine, die uns einen beliebigen String mit 64 Zeichen pro Zeile auf den Bildschirm ausgibt.

```
5    CLEAR 63487: LOAD*"m";1;"64MC" CODE
10   LET pr64=9700
20   LET xpos=10
30   LET ypos=0
50   CLS
60   INPUT "Bitte Text eingeben",,z$
70   GO SUB pr64
80   PAUSE 0
90   GO TO 50
9700 PRINT AT xpos,ypos;CHR$8;
9710 FOR i=1 TO LEN z$ STEP 2
9720 POKE 23607,251
9730 PRINT z$(i);
9740 IF i=LEN z$ THEN GO TO 9790
9750 PRINT CHR$ 8;
9760 POKE 23607,247
9770 PRINT OVER 1;z$(i+1);
9780 NEXT i
9790 POKE 23607,60
9800 RETURN
```

So, jetzt fehlen uns nur noch die beiden Zeichensätze und wir können loslegen. Die Zeichen zu definieren, ist natürlich etwas mühselig. Wer das Programm Tasword II be-

sitzt, kann sich die Sache einfach machen, da hier ein Satz bereits definiert ist. Er steht, wenn man das Programm geladen hat, an den Speicherplätzen 61184 bis 62079 und kann, wenn man ins Basic zurückkehrt, leicht geSAVEd werden mit

SAVE "64MC" CODE 61184,896 bzw.
SAVE *"m";1;"64MC" CODE 61184,896

Danach kann er mit LOAD "64MC" CODE 63488,896 an die von uns benutzten Speicherplätze geladen werden. Wer das Programm Tasword II nicht besitzt, muß in den sauren Apfel beißen und den Code eintippen. Das Programm zur Herstellung dieses Zeichensatzes sieht dann folgendermaßen aus:

```
10    CLEAR 63487
20    RESTORE
32    DATA 0,0,0,0,0,0,0,0
33    DATA 0,2,2,2,2,0,2,0
34    DATA 0,5,5,0,0,0,0,0
35    DATA 0,5,7,5,5,7,5,0
36    DATA 0,2,7,4,7,1,7,2
37    DATA 0,4,4,1,2,4,1,1
38    DATA 0,2,5,2,6,11,15,0
39    DATA 0,2,4,0,0,0,0,0
40    DATA 0,2,4,4,4,4,2,0
41    DATA 0,4,2,2,2,2,4,0
42    DATA 0,0,5,2,7,2,5,0
43    DATA 0,0,2,2,7,2,2,0
44    DATA 0,0,0,0,0,2,2,4
45    DATA 0,0,0,0,7,0,0,0
46    DATA 0,0,0,0,0,6,6,0
47    DATA 0,1,1,2,2,4,4,0
48    DATA 0,7,5,5,5,5,7,0
49    DATA 0,2,6,2,2,2,7,0
50    DATA 0,2,5,1,2,4,7,0
51    DATA 0,6,1,6,1,1,6,0
52    DATA 0,1,3,5,5,7,1,0
53    DATA 0,7,4,6,1,1,6,0
```

```
54    DATA 0,3,4,6,5,5,2,0
55    DATA 0,7,1,2,2,4,4,0
56    DATA 0,7,5,2,5,5,7,0
57    DATA 0,2,5,5,3,1,6,0
58    DATA 0,0,0,2,0,0,2,0
59    DATA 0,0,2,0,0,2,2,4
60    DATA 0,0,1,2,4,2,1,0
61    DATA 0,0,0,7,0,7,0,0
62    DATA 0,0,4,2,1,2,4,0
63    DATA 0,2,5,1,2,0,2,0
64    DATA 3,4,2,5,2,1,6,0
65    DATA 0,2,5,5,7,5,5,0
66    DATA 0,6,5,6,5,5,6,0
67    DATA 0,2,5,4,4,5,2,0
68    DATA 0,6,5,5,5,5,6,0
69    DATA 0,7,4,6,4,4,7,0
70    DATA 0,7,4,7,4,4,4,0
71    DATA 0,2,5,4,7,5,2,0
72    DATA 0,5,5,7,5,5,5,0
73    DATA 0,7,2,2,2,2,7,0
74    DATA 0,1,1,1,5,5,2,0
75    DATA 0,5,5,6,6,5,5,0
76    DATA 0,4,4,4,4,4,7,0
77    DATA 0,5,7,7,7,5,5,0
78    DATA 0,7,5,5,5,5,5,0
79    DATA 0,2,5,5,5,5,2,0
80    DATA 0,6,5,5,6,4,4,0
81    DATA 0,7,5,5,5,7,7,1
82    DATA 0,7,5,5,6,6,5,0
83    DATA 0,3,4,2,1,1,6,0
84    DATA 0,7,2,2,2,2,2,0
85    DATA 0,5,5,5,5,5,7,0
86    DATA 0,5,5,5,5,5,2,0
87    DATA 0,5,7,7,7,7,2,0
88    DATA 0,5,5,2,2,5,5,0
89    DATA 0,5,5,5,2,2,2,0
90    DATA 0,7,1,2,2,4,7,0
91    DATA 0,5,2,5,7,5,5,0
92    DATA 0,5,0,7,5,5,7,0
93    DATA 0,5,0,5,5,5,7,0
```

```
94   DATA 0,2,7,2,2,2,2,0
95   DATA 0,0,0,0,0,0,0,15
96   DATA 0,2,5,4,15,4,15,0
97   DATA 0,0,6,1,7,5,7,0
98   DATA 0,4,4,6,5,5,6,0
99   DATA 0,0,3,4,4,4,3,0
100  DATA 0,1,1,3,5,5,3,0
101  DATA 0,0,2,5,6,4,3,0
102  DATA 0,3,4,6,4,4,4,0
103  DATA 0,0,3,5,5,3,1,6
104  DATA 0,4,4,6,5,5,5,0
105  DATA 0,2,0,6,2,2,7,0
106  DATA 0,1,0,1,1,1,5,2
107  DATA 0,4,5,6,6,5,5,0
108  DATA 0,4,4,4,4,4,3,0
109  DATA 0,0,5,7,7,7,5,0
110  DATA 0,0,6,5,5,5,5,0
111  DATA 0,0,2,5,5,5,2,0
112  DATA 0,0,6,5,5,6,4,4
113  DATA 0,0,3,5,5,3,1,1
114  DATA 0,0,3,4,4,4,4,0
115  DATA 0,0,3,4,2,1,6,0
116  DATA 0,2,7,2,2,2,1,0
117  DATA 0,0,5,5,5,5,7,0
118  DATA 0,0,5,5,5,5,2,0
119  DATA 0,0,5,7,7,7,2,0
120  DATA 0,0,5,5,2,5,5,0
121  DATA 0,0,5,5,5,3,1,6
122  DATA 0,0,7,1,2,4,7,0
123  DATA 0,5,0,6,1,7,7,0
124  DATA 0,5,0,2,5,5,2,0
125  DATA 0,5,0,0,5,5,7,0
126  DATA 0,6,5,6,5,5,6,4
127  DATA 6,8,11,10,11,8,9,6
128  DATA 15,9,9,9,9,9,9,15
129  DATA 3,3,3,3,0,0,0,0
130  DATA 12,12,12,12,0,0,0,0
131  DATA 15,15,15,15,0,0,0,0,
132  DATA 0,0,0,0,3,3,3,3
133  DATA 3,3,3,3,3,3,3,3
```

```
134  DATA 12,12,12,12,3,3,3,3
135  DATA 15,15,15,15,3,3,3,3
136  DATA 0,0,0,0,12,12,12,12
137  DATA 3,3,3,3,12,12,12,12
138  DATA 12,12,12,12,12,12,12,12
139  DATA 15,15,15,15,12,12,12,12
140  DATA 0,0,0,0,15,15,15,15
141  DATA 3,3,3,3,15,15,15,15
142  DATA 12,12,12,12,15,15,15,15
143  DATA 15,15,15,15,15,15,15,15

200  DATA 9999
250  LET n=63487
300  READ x
310  IF x>500 THEN GO TO 400
320  LET n=n+1
330  POKE n,x
390  GO TO 300
```

Wir können in diesem Programm auch gleich den zweiten Schriftsatz definieren. Nein, keine Angst, dazu brauchen wir nicht wieder so viele Data Anweisungen zu tippen. Wir können diese Zeichencodes nämlich einfach aus den vorhandenen Data Anweisungen berechnen. Da beim zweiten Code immer die linke Hälfte der Matrix benutzt wird, ist jede Zahl einfach mit 16 zu multiplizieren, und schon sind wir fertig. Geben wir also ein:

```
340  POKE n+1024,16*x
```

und weiter

```
400  SAVE "64MC" CODE 63488,1921
```

oder die entsprechende Anweisung, um auf Microdrive zu sichern.

Jetzt geben Sie bitte ein RUN auf dieses Programm und sichern die beiden Zeichensätze. Dann können Sie das vorhin geschriebene Programm eintippen oder laden und sind fertig.

Die Möglichkeiten, die Routine 'pr64' zu nutzen und je nach Bedarf umzuschreiben, sind vielfältig. So können Sie zum Beispiel auch jede Eingabe mit dieser Schrift durchführen. Lassen Sie Ihrer Phantasie freien Lauf.

Mastermind

Wir wollen nun das erste kleine Spiel programmieren. Es heißt Mastermind. Bei diesem Spiel 'denkt' sich der Spectrum aus den Zahlen von 1 bis 9 fünf verschiedene Zahlen in beliebiger Reihenfolge aus. Der Spieler soll versuchen, sowohl die Zahlen, als auch die Reihenfolge der Zahlen mit möglichst wenig Versuchen zu erraten. Nach jedem Tip, den der Spieler abgegeben hat, sagt der Spectrum einmal, wieviele Zahlen überhaupt richtig geraten wurden, zum zweiten gibt er an, wieviele der richtigen Zahlen an der richtigen Stelle stehen.

Ehe wir mit der Programmierung beginnen, wollen wir uns einen Plan des Programms machen, also so etwas ähnliches wie ein Flußdiagramm. Die einzelnen Aufgaben sind die folgenden:

INITIALISIEREN, d. h. Speicherplatz bereitstellen
Bildschirm löschen usw.

'AUSDENKEN' DER ZAHLEN

BILDSCHIRM AUFSETZEN

EINGABE DES TIPS

EINGABE AUF RICHTIGKEIT PRÜFEN

AUSGABE DES TIPS

ANZAHL RICHTIGER POSITIONEN PRÜFEN

ANZAHL RICHTIGER ZAHLEN PRÜFEN

AUSGABE DER TREFFER

PRÜFEN OB LÖSUNG GEFUNDEN
wenn ja, dann Kommentar geben und ENDE

wenn nein, dann prüfen ob zu viele Versuche
wenn nein, dann zu EINGABE DES TIPS
wenn ja, dann Kommentar und ENDE

KOMMENTARE

ENDE

Beginnen wir mit dem ersten Punkt, dem Initialisieren.

```
4000 REM Mastermind
4010 BORDER 6: PAPER 6: INK 0: CLS
```

An Speicher benötigen wir zwei Felder, die jeweils 5 Elemente haben. Das erste nennen wir 'a(5)' und speichern hier die vom Spectrum ausgesuchten Zahlen, das zweite, b(5), soll den eingegebenen Tip beinhalten. Also:

```
4020 DIM a(5): DIM b(5)
```

Nun führen wir noch einen Zähler ein, damit das Spiel nach spätestens 15 Versuchen abgebrochen werden kann.

```
4030 LET nn=0
```

Damit ist der erste Punkt erledigt und wir können den Spectrum veranlassen, sich 5 Zahlen auszudenken.

```
4040 FOR i=1 TO 5
4050 LET a(i)=1+INT (9*RND)
```

Bei der ersten Zahl brauchen wir natürlich nicht zu überprüfen, ob diese bereits vorhanden ist, also

```
4060 IF i=1 THEN GO TO 4100
4070 FOR k=1 TO i-1
4080 IF a(i)=a(i-k) THEN GO TO 4050
4090 NEXT k
4100 NEXT i
```

Nun stehen im Feld a(5) fünf verschiedene Zahlen, wie gewünscht. Damit können wir dazu übergehen, den Bildschirm zu beschriften:

```
4200 PRINT "    MASTERMIND  (5 aus 9)"
4210 PRINT AT nn+3,0;"N        TIP      Pos OK   Num OK"
```

Und nun die Eingabe des Tips:

```
4220 INPUT (Tip (z.B. 12345 ENTER) ";LINE b$
```

Jetzt überprüfen wir diese Eingabe auf ihre Richtigkeit.

```
4230 IF LEN b$<>5 THEN GO TO 4220
4232 LET fehler=0
4234 FOR i=1 to 5
4236 IF CODE b$(i)<49 OR CODE b$(i)>57 THEN LET fehler=1
4238 NEXT i
4239 IF fehler=1 THEN GO TO 4220
```

Damit kann der Tip ausgegeben werden. Es ist eigentlich nicht notwendig, den Inhalt des Strings b$ in das Feld zu bringen. Bei diesem Programm haben wir aber sicher keine Probleme mit dem Speicherplatz und können dies getrost tun.

```
4240 PRINT AT nn+4,0;nn+1
4250 FOR i=1 TO 5
4260 LET b(i)=VAL b$(i)
4270 PRINT AT nn+4,2*i+2;b(i)
4280 NEXT i
4290 LET nn=nn+1
```

Nun überprüfen wir, ob die Lösung gefunden ist, bzw. wieviele Zahlen und Positionen richtig geraten sind. Zunächst werden zwei Zähler zu Null gesetzt:

```
4300 LET r1=0: LET r2=0
```

und eine Schleife von 1 bis 5 gemacht

```
4310 FOR i=1 TO 5
4320 IF a(i)=b(i) THEN LET r1=r1+1
```

Die Zahlen an der richtigen Stelle werden gezählt. Nun wird auf richtige Zahlen, egal an welcher Stelle, geprüft.

```
4330 FOR k=1 TO 5
4340 IF a(i)=b(k) THEN LET r2=r2+1
4350 NEXT k
4360 NEXT i
```

Damit sind die gesuchten Größen in r1 und r2 und können ausgegeben werden.

```
4370 PRINT AT nn+3,18;r1
4380 PRINT AT nn+3;25;r2
```

Wenn alle fünf Zahlen an der richtigen Stelle stehen, muß ein Kommentar ausgegeben werden.

```
4400 IF r1=5 THEN GO TO 4500
```

Wenn 15 Versuche vorbei sind, soll abgebrochen werden.

```
4410 IF nn>15 THEN GO TO 4600
```

sonst wieder zurück zur Eingabe

```
4420 GO TO 4220
```

Nun gehen wir an die Kommentare:

```
4500 PRINT: PRINT "In ";nn;" Versuchen gewonnen"
4510 IF nn<=4 THEN PRINT " Sehr gut"
4520 IF nn=5 THEN PRINT "Beinahe ein guter Spieler"
4530 IF nn=6 THEN PRINT "Recht ordentlich"
4540 IF nn=7 THEN PRINT "Fast gut, weiterueben!"
```

```
4550 IF nn>7 AND nn<=10 THEN PRINT "Durchschnitt"
4560 IF nn>10 THEN PRINT "Das hätte besser sein muessen!"
4570 GO TO 4900

4600 PRINT: PRINT "Sie sind heute nicht gut!"
```

Und zum Schluß noch die Abfrage, ob ein weiteres Spiel gewünscht wird

```
4900 PRINT FLASH 1;"Noch ein Spiel? J/N"
4910 PAUSE 0
4920 IF INKEY$="j" THEN GO TO 4000
4930 CLS : STOP
```

Damit sind wir fertig. Nun sichern Sie bitte das Programm mit 'LINE 1' und geben RUN ein. Viel Spaß beim Raten!

Übrigens ist das Programm nicht so knapp wie möglich geschrieben, damit es etwas übersichtlicher bleibt. Sie sind natürlich eingeladen, das Programm zu verbessern, so wie jedes Programm in diesem Buch. Es sollen hier zwar lauffähige Programme gezeigt werden, das soll Sie aber nicht davon abhalten, Ihre eigene Kreativität walten und schalten zu lassen.

Zum Abschluß dieses Kapitels werden noch drei Listings von Programmen angegeben, die nicht näher erläutert werden. Die Programme sind relativ kurz und können mit gutem Willen relativ schnell verstanden werden. Das erste gibt Ihnen eine 'Tagesschau-Uhr' auf dem Bildschirm aus. Das Programm ist in den Grundzügen im Sinclair Handbuch beschrieben. Hier ist es vervollständigt. Dies Programm läßt sich natürlich auch als Unterroutine in anderen Programmen verwenden.

Genauso ist es mit dem Programm 'Kalender', welches danach gelistet ist. Das Programm gestattet Ihnen, jeden beliebigen Monat mit Wochentagen auf dem Schirm zu zeigen.

Das dritte Programm ist der bei einigen Menschen so beliebte Biorhythmus. Natürlich hat der Biorhythmus nichts mit der Realität zu tun, aber man kann ja nie wissen.....

```
   1 BORDER 0: PAPER 0: INK 6: CLS
  10 LET u$="1111"
  20  GO TO 2000
  50 INPUT "Uhrzeit (z.B. 0845)"; LINE u$
  55 IF LEN u$<>4 THEN  GO TO 50
  60 LET fehler=0
  70 FOR i=1  TO 4
  75 IF CODE u$(i)<48 OR CODE u$(i)>57 THEN  LET fehler=1
  80 NEXT i
  85 IF fehler=1 THEN  GO TO 50
  90 IF VAL u$>2400 THEN  GO TO 50
  95 RETURN
2000 CLS : REM uhr
2100 LET e=50*(3600*VAL u$(1 TO 2)+60*VAL u$(3 TO 4))
2110 LET ss=65536: LET pp=23672
2120 LET a1=INT (e/ss)
2130 LET a2=INT ((e-a1*ss)/256)
2140 LET a3=e-a1*ss-a2*256
2150 POKE pp+2,a1: POKE pp+1,a2: POKE pp,a3
2200 CLS
2210 FOR n=1 TO 12: PRINT AT 10-10*COS (n/6*PI),16+10*SIN (n/6*PI);n: NEXT n
2220 DEF FN t()=INT ((ss*PEEK (pp+2)+256*PEEK (pp+1)+PEEK pp)/50)
2230 LET t1=FN t(): LET ts=t1
2240 LET tm=INT (t1/60): LET th=tm/60
2250 LET am=tm/30*PI: LET ah=th/6*PI
2260 LET mx=72*SIN am: LET my=72*COS am
2270 LET hx=40*SIN ah: LET hy=40*COS ah
2300 PLOT 132,90: DRAW  OVER 1;hx,hy
2310 PLOT 130,90: DRAW  OVER 1;mx,my
2320 LET a=t1/30*PI
2330 LET sx=56*SIN a: LET sy=56*COS a
2340 PLOT 131,91: DRAW  OVER 1;sx,sy
2350 LET t=FN t()
2360 IF t<=t1 THEN  GO TO 2350
2370 PLOT 131,91: DRAW  OVER 1;sx,sy
2380 IF INKEY$="u" THEN  GO SUB 50: GO TO 2000
2400 IF t/60<>INT (t/60) THEN  GO TO 2450
2410 PLOT 130,90: DRAW  OVER 1;mx,my
2420 PLOT 132,90: DRAW  OVER 1;hx,hy
2430 LET tm=tm+1: LET th=th+1/60
2440>LET ts=t:LET t1=t
2445 GO TO 2250
2450 LET t1=t
2500 GO TO 2320
```

```
  1 BORDER 0: PAPER 0: INK 6: CLS
 10 GO TO 100
 20 LET y=0
 22 FOR j=1 TO LEN m$: IF m$(j)="*" THEN  LET y=y+1
 23 IF y=m THEN  GO TO 25
 24 NEXT j
 25 FOR y=1 TO 9: IF m$(j+y)<>"*" THEN  NEXT y
 28 LET t$=m$(j+1 TO j+y-1)
 29 RETURN
 30 INPUT "Tag = "; LINE a$: IF a$="" THEN  GO TO 30
 32 IF VAL a$<10 THEN  LET a$=" "+a$
 35 LET tag=VAL a$
 40 RETURN
100 LET m$="*Januar*Februar*Maerz*April*Mai*Juni*Juli*August*September*Oktober*
November*Dezember*"
110 LET n$=" 1 2 3 4 5 6 7 8 910111213141516171819202122232425262728293031"
120 LET d$="SonMonDieMitDonFreSam"
130 LET c$="303232332323"
140 LET b=0: LET c=0
200 INPUT "Jahr? (enter=1984) ";a$
202 IF a$="" THEN  LET ty=1984: LET a$="1984"
203 IF VAL a$<1760 THEN  GO TO 200
204 IF VAL a$>9999 THEN  GO TO 200
208 IF a$<>"" THEN  LET ty=VAL a$
210 INPUT "Monat ?(1 = Jan etc) ";a$: IF a$="" THEN  GO TO 210
215 LET m=VAL a$
220 GO SUB 20
230 GO SUB 30
400 CLS : LET ly=0
402 IF ty/4<>INT (ty/4) THEN  GO TO 420
404 LET ly=1
410 IF ty/100=INT (ty/100) AND ty/400<>INT (ty/400) THEN  LET ly=0
420 LET n=CODE c$(m)-20
422 IF m=2 AND ly THEN  LET n=n+1
424 LET y=0
426 FOR j=1 TO m: LET y=y+CODE c$(j)-20: NEXT j
430 IF ly AND m>1 THEN  LET y=y+1
435 LET y=y+1-n+(ty-1)*365+INT ((ty-1)/4)-INT ((ty-1)/100)+INT ((ty-1)/400)
440 LET y=y-7*INT (y/7)
450 CLS : PRINT  INK 3;ty;TAB (31-LEN t$)/2;t$;TAB 28;ty
460 FOR j=1 TO LEN d$ STEP 3: PRINT AT j/3*2+2,0;d$(j TO j+2);TAB 29;d$(j TO j+
2): NEXT j
```

```
500 LET p$=n$( TO n*2)
502 FOR j=1 TO y: LET p$="  "+p$: NEXT j
520 LET date=0: LET x=5: LET y=1
530 FOR j=1 TO LEN p$ STEP 2: LET y=y+2: IF y<16 THEN  GO TO 560
540 LET y=3: LET x=x+4
560 IF p$(j TO j+1)=a$ THEN  FLASH 1
565 PRINT AT y,x; INK 4;p$(j TO j+1)
566 FLASH 0
570 NEXT j
580 PRINT AT 19,0;"Naechster Monat    Letzter Monat"
581 PRINT AT 20,0;"Datum"
590 PRINT AT 20,0;"Datum   (1. Buchstaben eingeben)"
600 LET f$=INKEY$
610 IF f$="d" THEN  GO TO 200
650 IF f$<>"n" THEN  GO TO 670
660 LET m=m+1: LET a$="99": IF m=13 THEN  LET ty=ty+1: LET m=1
661 IF ty>9999 THEN  LET ty=9999
665 GO SUB 20
669 GO TO 400
670 IF f$<>"l" THEN  GO TO 680
672 LET m=m-1: IF m=0 THEN  LET m=12: LET ty=ty-1: IF ty<1760 THEN  LET ty=1760
673 LET a$="99"
674 GO SUB 20
676 GO TO 400
680 GO TO 600
```

```
  1 BORDER 6: PAPER 6: INK 0: CLS
 10 LET m$="303232332323"
 20 PRINT TAB 6; INVERSE 1;"B I O R H Y T H M U S"
 30 INPUT "Name= ";n$
 40 PRINT AT 2,0;"von ";n$
 50 INPUT "Geburtsjahr";j$
 55 IF VAL j$<=1900 THEN  GO TO 50
 60 PRINT "geb. : ";j$;"-";
 70 INPUT "Geburtsmonat (1-12) ";x$
 80 PRINT x$;"-";
 90 INPUT "Geburtstag = ";t$
100 PRINT t$
110 LET q=0
120 FOR i=1 TO VAL t$-1
130 LET q=q+CODE m$(i)-20
140 NEXT i
145 LET y=VAL j$
150 IF q>59 AND y/4=INT (y/4) THEN  LET q=q+1
160 LET q=q+VAL t$+INT (365.25*(y-1))
200 INPUT "Datum:Jahr = ";k$
205 IF k$<j$ THEN  GO TO 200
210 PRINT "Datum: ";k$;"-";
220 INPUT "Monat = ";l$
230 PRINT l$;"-";
240 INPUT "Tag ";y$
250 PRINT y$
300 LET r=0
310 FOR i=1 TO VAL l$-1
320 LET r=r+CODE m$(i)-20
330 NEXT i
335 LET y=VAL k$
340 IF r>59 AND y/4=INT (y/4) THEN  LET r=r+1
350 LET r=r+VAL y$+INT (365.25*(y-1))
400 LET z=r-q
600 LET amp=1.5
610 PRINT AT 8,0;"________________________________"
620 PRINT AT 13,0;"________________________________"
630 PRINT AT 18,0;"________________________________"
640 GO SUB 1100
650 PRINT AT 7,0;"physisch"
660 PRINT AT 12,0;"emotional"
670 PRINT AT 17,0;"intellektuell"
```

```
 700 LET phy=z-23*INT (z/23)
 710 LET emo=z-28*INT (z/28)
 720 LET int=z-33*INT (z/33)
 750 LET p=23
 760 LET n=phy
 770 LET k=104
 780 GO SUB 1000
 800 LET p=28: LET n=emo: LET k=64: GO SUB 1000
 810 LET p=33: LET n=int: LET k=24: GO SUB 1000
 900 INPUT "Kopie auf den Drucker? J/N"; LINE q$: IF q$="j" THEN  COPY
 910 INPUT "nocheinmal? J/N"; LINE q$: IF q$="j" THEN  GO TO 1
 920 CLS : STOP
1000 REM Kurvenmalen
1010 FOR j=0 TO 255
1020 PLOT j,amp*SIN ((j/4+(8*n))*PI/p)*10+k
1030 NEXT j
1040 RETURN
1100 FOR i=0 TO 30 STEP 5
1110 LET tag=VAL y$+i
1120 IF tag>CODE m$(VAL l$)-20 THEN  LET tag=tag-CODE m$(VAL l$)+20
1130 PRINT AT 9,i;tag
1140 PRINT AT 14,i;tag
1150 PRINT AT 19,i;tag
1160 NEXT i
1170 RETURN
```

KAPITEL 5:

E I N S P I E L F Ü R H E L L E K Ö P F C H E N: S Y M B O L R A T E N

Erläuterung des Spiels

Nachdem wir uns bislang mehr mit grundsätzlicheren Dingen und allerlei Tips,Tricks und kleineren Routinen und Programmen auseinandergesetzt haben, wollen wir nun richtig zur Sache gehen und ein etwas umfangreicheres Spielprogramm erstellen.

Wie die Überschrift zu diesem Kapitel ja schon andeutet, handelt es sich nicht um ein Kriegsspiel. Es kommt auch nicht besonders auf die Reaktionsschnelligkeit an. Vielmehr wird das Spiel das logische Denken trainieren. Aber keine Angst, es macht trotzdem Spaß!

Die Aufgabe besteht darin, einen Satz von miteinander verknüpften Gleichungen zu knacken, oder, auf deutsch, ein System korrelierter Gleichungen zu lösen. Was heißt das? Schreiben wir einfach einmal eine Gleichung hin:

$$5 + 6 = 11$$

Nun schreiben wir eine zweite Gleichung hin:

$$14 + 7 = 21$$

Ziehen wir nun noch einen Strich unter die beiden Gleichungen und addieren die untereinanderstehenden Zahlen, so ergibt sich folgendes Bild:

```
 5 +  6 = 11
14 +  7 = 21
```

19 + 13 = 32

Was ist das Interessante hieran? Nun, wenn das Programm fertig ist, dann 'denkt' der Spectrum sich ein solches Gleichungssystem (und etwas kompliziertere) aus, aber er wird uns nicht die Zahlen selbst zeigen, sondern jede Zahl von 0 bis 9 durch ein bestimmtes Symbol darstellen. Die Aufgabe des Spielers besteht dann darin, zu jedem Symbol die entsprechende Zahl zu finden.

Da wir zum Aufsetzen der Gleichungen und bei der Zuordnung der Symbole zu den Zahlen den Zufallsgenerator des Spectrums in Anspruch nehmen, wird kein Spiel dem anderen gleichen. Anfangs wird man wohl etwas Hilfestellung vom Computer haben wollen, je öfter man spielt, desto weniger wird dies allerdings der Fall sein.

Entwicklung des Programms

Folgende Routinen werden für dieses Programm benötigt:

1. Titelseite mit Erklärung des Spiels zeigen

2. Speicherplatz für alle möglichen Felder (Vektoren) reservieren

3. Aufstellen des Gleichungssystems

4. Zuordnung von Symbolen zu Zahlen

5. Aufsetzen des Bildschirms

6. Eingabe von Tips, Änderungen, und Ausgabe von Hilfen des Computers

7. Überprüfung, ob die Lösung gefunden wurde

8. Ende des Spiels

Beginnen wir mit dem ersten Punkt, dem Aufsetzen der Titelseite.

```
100 PAPER 6: BORDER 3: INK 0
200 CLS: PRINT TAB 9;"SYMBOL RATEN"
210 PRINT "Es gilt, ein mathematisches Pro-blem zu loesen
     und herauszufin-"
220 PRINT "den, welches Symbol welcher Zahl(von  0  -  9)
    entspricht."
230 PRINT "Verschiedene Symbole entsprechenverschiedenen
    Zahlen."
```

Bitte achten Sie beim Eingeben dieses Textes auf die richtige Eingabe von Leerzeichen, sodaß sich eine ansprechende Titelseite ergibt.

Wir müssen nun festlegen, wie das Spiel gespielt werden soll, d. h. wie Eingaben gemacht werden können, damit wir dies auf der Titelseite angeben können.

Um die Sache übersichtlich zu gestalten, wollen wir unter dem durch Symbole dargestellten Gleichungssystem in einer waagerechten Linie alle Symbole zeigen. Unter dieser Reihe soll dann ein kleines blinkendes Zeichen durch Tastendruck nach rechts und links bewegbar sein, sodaß es unter jedes Symbol gesetzt werden kann. Gibt man dann eine Zahl ein, so erscheint diese über jedem entsprechenden Symbol in den Gleichungen.

Also fügen wir der Titelseite zu:

```
240 PRINT "Um  eine  Zahl  fuer ein  Symboleinzugeben, wi
    rd  das  blinkende"
250 PRINT "Sternchen (*) unter das entspre-chende Symbol
     gefuehrt und  die"
260 PRINT "Zahl eingegeben."
270 PRINT "L= * nach links,R= * nach rechts"
280 PRINT '"***  Weitere Moeglichkeiten  ***"
290 PRINT "D = Loescht alten Tip"
300 PRINT "V = Gibt den Wert fuer Symbol       ueber dem
    Sternchen"
310 PRINT "S = Loesung des Problems"
320 PRINT "N = Neues Spiel"
330 PRINT "E = Ende des Spiels"
340 PRINT "     "; FLASH 1;"Bitte eine Taste druecken!"
350 IF INKEY$="" THEN GO TO 350
360 BEEP .03,7
370 CLS : PRINT FLASH 1;AT 10,9;" Bitte warten ";AT 9,9;"
                  ";AT 11,9;"              "
```

Wenn Sie dies alles richtig eingegeben haben,sollte beim Eingeben von RUN der Bildschirm nicht überlaufen, d. h. es darf nicht die Abfrage SCROLL? erscheinen. Ist dies doch der Fall, überprüfen und verändern Sie den Text, um alles auf eine Seite zu packen.

Nun haben wir unser erstes richtige Problem zu lösen, wir müssen das Gleichungssystem aufsetzen. Insgesamt tauchen 9 verschiedene Zahlen auf, von denen wir nur einige frei wählen können, also durch den Zufallsgenerator des Spectrum erzeugen lassen können. Wir wollen diese Zahlen im Feld a(i) speichern. Das Ganze kann dann folgendermaßen verdeutlicht werden:

```
a(1) + a(2) = a(3)
  +      +      +
a(4) + a(5) = a(6)
  =      =      =
a(7) + a(8) = a(9)
```

Als größtmögliche Zahl soll 9999 auftauchen können, damit wir auf dem Bildschirm genügend Platz haben. Von diesen Zahlen sind a(1), a(2), a(4) und a(5) frei wählbar, allerdings a(2), a(4) und a(5) dürfen nicht zu groß ausfallen, damit nachher keine Zahl aus den Summen, zum Beispiel a(9), fünfstellig wird.

Wählen wir daher das folgende Schema:

```
1010 LET a(1)=1+INT (5000*RND)
1020 LET a(2)=1+INT (1000*RND)
1030 LET a(4)=1+INT (1000*RND)
1040 LET a(3)=a(1)+a(2)
1050 LET a(5)=1+INT ((8000-a(1))*RND)
```

und der Rest ist dann klar:

```
1060 LET a(6)=a(4)+a(5)
1070 LET a(7)=a(1)+a(4)
1080 LET a(8)=a(2)+a(5)
1090 LET a(9)=a(3)+a(6)
```

Wir können noch die Sicherheitsabfrage zufügen

```
1100 IF a(9)>9999 THEN GO TO 1010
```

In den Größen a(1) bis a(9) stehen nun die gewünschten Zahlen. Das ging einfacher als zunächst angenommen. Dann können wir eigentlich die Aufgabe noch etwas erschweren, indem wir nicht nur Additionen, sondern auch Subtraktionen zulassen. Zum Beispiel können wir auch das folgende System aufsetzen:

```
a(1) + a(2) = a(3)
 +      -      +
a(4) - a(5) = a(6)
 =      =      =
a(7) + a(8) = a(9)
```

Das geht ganz ähnlich dem ersten Beispiel, bringt aber sicher etwas Pfiff in das Problem. Lassen wir also den Zufallsgenerator entscheiden, welche Aufgabe gestellt wird, und setzen auch das zweite Problem auf.

Zunächst geben wir ein:

```
1000 LET s=0: IF RND<.5 THEN GO TO 1200
```

Damit kommt jede Möglichkeit gleich häufig vor. Weiter folgt bei Zeile 1200

```
1130 GO TO 1300
1200 LET s=1
```

Wenn s=0 ist, liegt also das erste, bei s=1 das zweite Problem vor. Verfahren wir jetzt wie beim ersten Problem, jedoch müssen wir aufpassen, daß keine der berechneten Zahlen kleiner als Null wird. Das geht so:

```
1210 LET a(1)=1+INT (5000*RND)
1220 LET a(2)=100+INT (900*RND)
1230 LET a(4)=100+INT (900*RND)
1235 LET a(3)=a(1)+a(2)
1240 LET a(7)=a(1)+a(4)
1245 LET a(5)=1+INT (1000*RND)
1250 IF a(5)>=a(2) OR a(5)>=a(4) THEN GO TO 1245
```

```
1260 LET a(8)=a(2)-a(5)
1265 LET a(6)=a(4)-a(5)
1270 LET a(9)=a(3)+a(6)
```

Wenn wir nun noch eingeben

```
50 DIM a(9)
```

haben wir diesen Teil des Programms geschafft.

Nun müssen wir uns die Symbole beschaffen. Auf dem Spectrum haben wir bereits acht Grafikzeichen, die wir benutzen können, das sind aber zwei zu wenig, um für jede der Zahlen 0 bis 9 eines zur Verfügung zu haben. Hier helfen die UDGs, also die USER DEFINED GRAFICS Symbole weiter. Wie im Handbuch beschrieben können wir uns zwei weitere Zeichen, wir wählen Dreiecke, einfach definieren. Wenn wir die Dreiecke spiegelsymmetrisch wählen, ist dies sogar noch weniger Aufwand:

```
20 DATA 128,192,224,240,248,252,254,255
30 FOR i=0 TO 7: READ xn: POKE USR "A"+i,xn:POKE USR "B"+
   7-i,xn: NEXT i
```

Nun speichern wir alle Symbole in das Feld a$ mit

```
80 LET a$="[illegible]"
```

Jede der 9 Zahlen a(i) soll jetzt durch Symbole dargestellt werden. Dazu generieren wir uns zunächst einmal die Zahlen von 0 bis 9 in 'zufälliger' Reihenfolge:

```
1300 FOR i=1 to 10
1310 LET z(i)=INT (10*RND)
```

Die erste Zahl kann beliebig sein, bei allen weiteren müssen wir nachschauen, ob die gewürfelte Zahl nicht schon vorhanden ist.

```
1320 IF i=1 THEN GO TO 1360
```

```
1330 FOR k=1 TO i-1
1340 IF ABS (z(k)-z(i))<=0.5 THEN GO TO 1310
1350 NEXT k
1360 LET z$(i)=a$(z(i+1))
1370 NEXT i
```

In dem Feld z(i) stehen nun die Zahlen von 0 bis 9 in zufälliger Reihenfolge und im String z$(i) die entsprechenden Grafikzeichen. z$(i) ist also ein Feld wie a$, nur daß in a$ die Zeichen immer in der gleichen Reihenfolge stehen, in z$ dagegen in einer zufällig gewählten.

Als nächste Aufgabe müssen wir jedes dieser Zeichen an die richtige Stelle setzen, sodaß jede Ziffer der Zahlen a(i) durch ein Symbol ersetzt wird. Dabei müssen wir aufpassen, daß die Symbole in den Strings, welche die Zahlen darstellen, rechtsbündig erscheinen. Das hört sich etwas kompliziert an, soll aber nur bedeuten, daß 'Einer' unter 'Einern' stehen, 'Zehner' unter 'Zehnern' und so weiter.

Um dieses zu erreichen, formen wir die Zahlen a(i) zunächst in Strings um. Diese Strings kopieren wir anschliessend in 'rechtsbündige' Strings w$ und haben erreicht, was wir wollen.

```
1380 FOR i=1 TO 9
1390 LET q$(i)=STR$ a(i)
1400 LET l=LEN (STR$ (a(i)))
1410 FOR k=1 TO l
1420 LET w$(i,5-k)=z$(VAL (q$(i,l+1-k))+1)
1430 NEXT k
1440 NEXT i
```

Die benutzten Felder müssen wir noch dimensionieren:

```
60   DIM z(10): DIM z$(10)
70   DIM q$(9,5): DIM w$(9,4)
```

Nach diesen Vorarbeiten können wir nun die Aufgabe auf dem Bildschirm darstellen.

```
2000 CLS : PRINT "██████P R O B L E M███████"
2010 PRINT AT 17,0;"████████████████████████████████"
2020 FOR i=1 TO 10: PRINT AT 19,3*i-1;z$(z(i)+1): NEXT i
2030 PRINT AT 13,0;"█"
2040 PRINT AT 21,0;"█"
2050 FOR i=0 TO 13: PRINT AT i,0;"█";AT i,31;"█": NEXT i
2060 FOR i"17 TO 21: PRINT AT i,0;"█";AT i,31;"█": NEXT i
2070 FOR i=1 TO 9
2080 LET k=0: LET n=0
2090 IF i=2 OR i=5 OR i=8 THEN LET k=1
2100 IF i=3 OR i=6 OR i=9 THEN LET k=2
2110 IF a(i)>9 THEN LET n=1
2120 IF a(i)>99 THEN LET n=2
2130 IF a(i)>999 THEN LET n=3
2140 LET y(i)=6-2*n+10*k
2150 LET f(i)=10*k
2160 LET LET x(i)=2+4*INT ((i-1)/3)
2170 NEXT i
2180 FOR i=1 TO 9
2190 FOR k=1 TO 4
2200 PRINT AT x(i)+1,f(i)+2*k;w$(i,k)
2210 NEXT k
2220 NEXT i
```

Nun haben wir alle Symbole an der richtigen Stelle und müssen nur noch die Plus- und Minuszeichen einfügen.

```
2230 PRINT AT 3,10;"+";AT 3,20;"="
2240 PRINT AT 7,10;"+";AT 7,20;"="
2250 PRINT AT 11,10;"+";AT 11,20;"="
2260 PRINT AT 5,6;"+";AT 5,16;"+";AT 5,26;"+"
2270 PRINT AT 9,6;"=";AT 9,16;"=";AT 9,26;"="
```

Für den Fall, daß auch Subtraktionen dabei sind, müssen wir hinzufügen

```
2280 IS s>=1 THEN PRINT AT 7,10;"-";AT 5,16;"-"
```

Das Bild ist so aufgebaut, daß noch Platz für die Erklärungen ist.

```
3000 REM Cursor
3010 LET x=21
3020 LET y=2
3100 PRINT FLASH 1;AT x,y;"*"
3200 PRINT AT 14,0;"L = * links          R = * rechts"
3210 PRINT AT 15,0;"D=loesche,           V=Wert geben"
3220 PRINT AT 16,0;"S=Loesung, N=Neues Spiel, E=Ende"
3300 PAUSE 0: LET m$=INKEY$
3310 IF m$="r" THEN BEEP .1,1: GO TO 3700
3320 IF m$="[illegible]" THEN BEEP .1,1: GO TO 3800
3330 IF m$="e" THEN GO TO 6000
3340 IF m$="n" THEN GO TO 1
3350 IF m$="v" THEN BEEP .3,5: GO TO 7000
3360 IF m$="s" THEN GO TO 4000
3370 IF m$="d" THEN BEEP .3,5: GO TO 5000
3380 IF CODE m$<58 AND CODE m$>47 THEN GO TO 5000
3390 BEEP .05,5: BEEP .1,1
3400 GO TO 3300
```

Damit sind die restlichen Routinen festgelegt. Ab Zeile 3700 wird der Cursor nach rechts, ab Zeile 3800 nach links verschoben. Ab Zeile 6000 wird das Spiel beendet, ab 7000 wird ein Wert vom Computer als Hilfestellung ausgegeben, ab 4000 die Lösung der Aufgabe geliefert und ab 5000 erfolgt die Eingabe bzw. das Löschen eines Tips.

```
3700 IF y>27 THEN GO TO 3300
3710 LET y=y+3
3720 PRINT AT x,y-3;" "
3730 PRINT FLASH 1;AT x,y;"*"
3740 FOR i=1 TO 50: NEXT i
3750 GO TO 3300

3800 IF y<3 THEN GO TO 3300
3810 LET y=y-3
3820 PRINT AT x,y+3;" "
```

```
3830 PRINT FLASH 1;AT x,y;"*"
3840 FOR i=1 TO 50: NEXT i
3850 GO TO 3300
```

Das Beenden des Spiels ist natürlich besonders einfach:

```
6000 PRINT AT 21,6;"Noch ein Spiel? J/N"
6010 PAUSE 0: LET m$=INKEY$
6020 IF m$="j" THEN GO TO 1
6030 CLS : STOP
```

Die Lösung des Problems läßt sich auch recht einfach ausgeben:

```
4000 REM Loesung
4010 FOR i=1 TO 20: BEEP .05,i: BEEP .05,-4: NEXT i
```

Diese Zeile ist zwar nicht nötig, macht sich aber ganz gut.

```
4020 FOR i= 1 TO 9
4030 FOR k=1 TO 4
4040 IF q$(i,k)=" " THEN GO TO 4060
4050 PRINT AT x(i),y(i)+2*k;q$(i,k)
4060 NEXT k
4070 NEXT i
4080 FOR i=1 TO 10
4090 PRINT AT 18,3*i-1;z(i)
4100 LET g(i)=z(i)
4110 NEXT i
4120 PRINT AT 14,0; PAPER 2,,,,,,
4130 PRINT FLASH 1;AT 15,6;"Das ist die Loesung"
4140 GO TO 6000
```

Zeile 4100 ist noch etwas verwunderlich. Wir wollen das Feld g(i) benutzen, um über die geratenen Symbole Buch zu führen, sodaß bei jeder Eingabe eines Tips überprüft werden kann, ob die Lösung bereits gefunden wurde. Dazu schreiben wir zu Beginn in alle g(i) eine Zahl größer als 9, zum Beispiel 22. Natürlich können Sie auch 432 hinein-

schreiben, falls Ihnen dies besser gefällt.

```
100 FOR i=1 TO 10: LET g(i)=22: NEXT i
```

In die Zeile 5000 wird gesprungen, wenn eine Eingabe gemacht wird oder eine Eingabe gelöscht werden soll.

```
5000 REM Tip eingeben
5010 LET j$=m$
5020 IF m$=d" THEN LET j$=" ": LET g(INT (y/3)+1)=22: GO
     TO 5200
5030 LET g(INT (y/3)+1)=VAL j$
5040 BEEP .04,1: BEEP .04,8
5200 FOR i=1 TO 9
5210 FOR k=1 TO 4
5300 IF w$(i,k)=z$(z(y/3)+1) THEN LET c$(i,k)=j$
5310 NEXT k
5320 NEXT i
```

Das ist alles ein wenig verschachtelt, aber es funktioniert. In c$ steht also, was bereits geraten wurde. Nun müssen wir dies nur noch auf den neuesten Stand bringen.

```
5500 PRINT AT 18,y;j$
5600 FOR i=1 TO 9
5610 FOR k=1 TO 4
5620 PRINT AT x(i),f(i)+2*k;c$(i,k)
5630 NEXT k
5640 NEXT i
```

Nun wird überprüft, ob die Lösung gefunden ist:

```
5800 FOR i=1 TO 10
5810 IF g(i)<>z(i) THEN GO TO 3300
5820 NEXT i
5830 PRINT AT 14,0; PAPER 2,,,,,,
5840 PRINT FLASH 1;AT 15,8;"G E W O N N E N"
```

In diesem Fall geht es automatisch bei 6000 mit der Frage ' Noch ein Spiel' weiter. Was uns noch fehlt, ist die

Routine, welche dem Spieler auf Wunsch eine Hilfe gibt und zu einem Symbol die entsprechende Zahl angibt.

```
7000 REM eine Zahl angeben
7010 PRINT AT 18,y;z(INT (y/3+1
7020 LET g(INT ((y/3)+1))=z(INT ((y/3)+1))
7030 FOR i=1 TO 9
7040 LET j=0: IF a(i)<1000 THEN LET j=1
7050 IF a(i)<100 THEN LET j=2
7060 IF a(i)<10 THEN LET j=3
7070 FOR k=1 TO 4
7080 IF w$(i,k)=z$(z(y/3)+1) THEN LET c$(i,k)=q$(i,k-j)
7090 PRINT AT x(i),y(i)+2*(k-j);c$(i,k)
7100 NEXT k
7110 NEXT i
7120 GO TO 3300
```

Die Anfangszeilen, in denen die Dimensionierung der Felder durchgeführt wird, müssen noch ergänzt werden. Sie sollten jetzt lauten:

```
50 DIM a(9): DIM x(9): DIM y(9): DIM f(9):DIM g(10)
60 DIM z$(10): DIM z(10)
70 DIM q$(9,5): DIM w$(9,4): DIM c$(9,4)
```

Zu guter Letzt fügen wir noch ein:

```
10 CLEAR: RESTORE: FOR i=1 TO 20: BEEP .5/i,i: NEXT i
```

und sind fertig. Zum Abschluß des Kapitels noch einmal das gesamte Listing ohne Unterbrechungen.

Nun viel Spaß beim Raten!

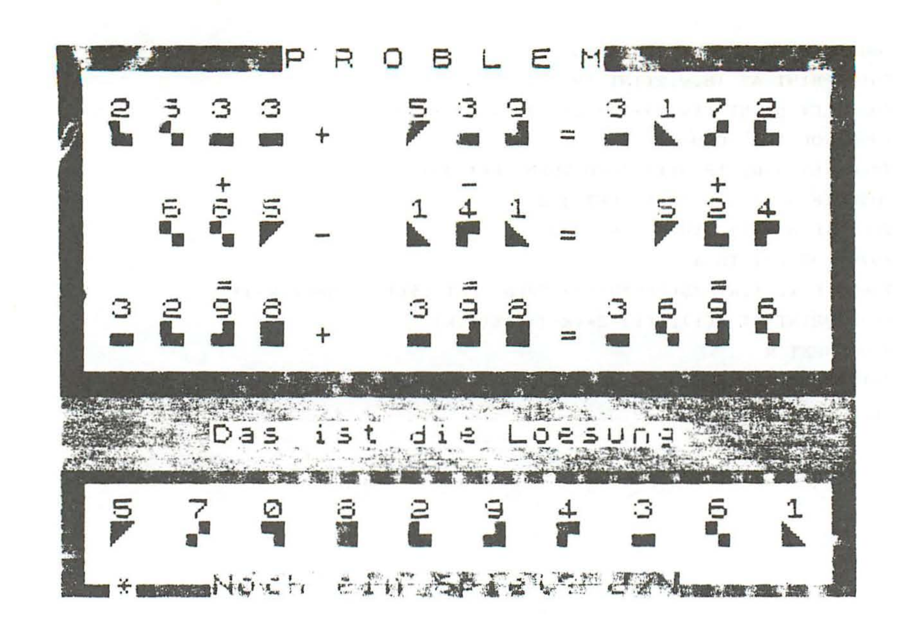
```
10 CLEAR : RESTORE : FOR i=1 TO 20: BEEP .5/i,i: NEXT i
20 DATA 128,192,224,240,248,252,254,255
30 FOR i=0 TO 7: READ xn: POKE USR "A"+i,xn: POKE USR "B"+7-i,xn: NEXT i
40 PAPER 6: BORDER 3: INK 0
50 DIM a(9): DIM x(9): DIM y(9): DIM f(9): DIM g(10)
60 DIM z$(10): DIM z(10)
70 DIM q$(9,5): DIM w$(9,4): DIM c$(9,4)
80 LET a$="??????????"
81 REM a$(1 bis 10) sind CHR$ 142,141,139,137,134,135,143,144,140,145
90 LET s=0
100 FOR i=1 TO 10: LET g(i)=22: NEXT i
200 CLS : PRINT "           SYMBOL RATEN  "
210 PRINT "Es gilt, ein mathematisches Pro-blem zu loesen  und herauszufin-"
220 PRINT "den, welches Symbol welcher Zahl(von 0 - 9) entspricht."
230 PRINT "Verschiedene Symbole entsprechenverschiedenen Zahlen."
240 PRINT "Um  eine  Zahl  fuer ein  Symboleinzugeben, wird  das  blinkende"
250 PRINT "Sternchen (*) unter das entspre-chende Symbol  gefuehrt und  die"
260 PRINT "Zahl eingegeben."
270 PRINT "L= * nach links,R= * nach rechts"
280 PRINT '"***  Weitere Moeglichkeiten  ***"
290 PRINT "D = loescht alten Tip"
300 PRINT "V = gibt den Wert fuer Symbol       ueber dem Sternchen"
310 PRINT "S = Loesung des Problems"
320 PRINT "N = Neues Spiel"
330 PRINT "E = Ende des Spiels"
340 PRINT "   "; FLASH 1;"Bitte eine Taste druecken!"
350 IF INKEY$="" THEN  GO TO 350
360 BEEP .03,7
370 CLS : PRINT  FLASH 1;AT 10,9;" Bitte warten ";AT 9,9;"              ";AT 11
,9;"              "
1000 IF RND<.5 THEN  GO TO 1200
1010 LET a(1)=1+INT (5000*RND)
1020 LET a(2)=1+INT (1000*RND)
1030 LET a(4)=1+INT (1000*RND)
1040 LET a(3)=a(1)+a(2)
1050 LET a(5)=1+INT ((8000-a(1))*RND)
1060 LET a(6)=a(4)+a(5)
1070 LET a(7)=a(1)+a(4)
1080 LET a(8)=a(2)+a(5)
1090 LET a(9)=a(3)+a(6)
1100 IF a(9)>9999 THEN  GO TO 1010
1130 GO TO 1300
```

```
1200 LET s=1
1210 LET a(1)=1+INT (5000*RND)
1220 LET a(2)=100+INT (900*RND)
1230 LET a(4)=100+INT (900*RND)
1235 LET a(3)=a(1)+a(2)
1240 LET a(7)=a(1)+a(4)
1245 LET a(5)=1+INT (1000*RND)
1250 IF a(5)>=a(2) OR a(5)>=a(4) THEN  GO TO 1245
1260 LET a(8)=a(2)-a(5)
1265 LET a(6)=a(4)-a(5)
1270 LET a(9)=a(3)+a(6)
1300 FOR i=1 TO 10
1310 LET z(i)=INT (10*RND)
1320 IF i=1 THEN  GO TO 1360
1330 FOR k=1 TO i-1
1340 IF ABS (z(k)-z(i))<=0.5 THEN  GO TO 1310
1350 NEXT k
1360 LET z$(i)=a$(z(i)+1)
1370 NEXT i
1380 FOR i=1 TO 9
1390 LET q$(i)=STR$ a(i)
1400 LET l=LEN (STR$ (a(i)))
1410 FOR k=1 TO l
1420 LET w$(i,5-k)=z$(VAL (q$(i,l+1-k))+1)
1430 NEXT k
1440 NEXT i
2000 CLS : PRINT "?????????P R O B L E M??????????"
2010 PRINT AT 17,0;"????????????????????????????????"
2020 FOR i=1 TO 10: PRINT AT 19,3*i-1;z$(z(i)+1): NEXT i
2030 PRINT AT 13,0;"????????????????????????????????"
2040 PRINT AT 21,0;"????????????????????????????????"
2050 FOR i=0 TO 13: PRINT AT i,0;"?";AT i,31;"?": NEXT i
2060 FOR i=17 TO 21: PRINT AT i,0;"?";AT i,31;"?": NEXT i
2070 FOR i=1 TO 9
2080 LET k=0: LET n=0
2090 IF i=2 OR i=5 OR i=8 THEN  LET k=1
2100 IF i=3 OR i=6 OR i=9 THEN  LET k=2
2110 IF a(i)>9 THEN  LET n=1
2120 IF a(i)>99 THEN  LET n=2
2130 IF a(i)>999 THEN  LET n=3
2140 LET y(i)=6-2*n+10*k
2150 LET f(i)=10*k
```

```
2160 LET x(i)=2+4*INT ((i-1)/3)
2170 NEXT i
2180 FOR i=1 TO 9
2190 FOR k=1 TO 4
2200 PRINT AT x(i)+1,f(i)+2*k;w$(i,k)
2210 NEXT k
2220 NEXT i
2230 PRINT AT 3,10;"+";AT 3,20;"="
2240 PRINT AT 7,10;"+";AT 7,20;"="
2250 PRINT AT 11,10;"+";AT 11,20;"="
2260 PRINT AT 5,6;"+";AT 5,16;"+";AT 5,26;"+"
2270 PRINT AT 9,6;"=";AT 9,16;"=";AT 9,26;"="
2280 IF s>=1 THEN  PRINT AT 7,10;"-";AT 5,16;"-"
3000 REM Cursor
3010 LET x=21
3020 LET y=2
3100 PRINT  FLASH 1;AT x,y;"*"
3200 PRINT AT 14,0;"L = * links          R = * rechts"
3210 PRINT AT 15,0;"D=loesche,           V=Wert geben"
3220 PRINT AT 16,0;"S=Loesung, N=Neues Spiel, E=Ende"
3300 PAUSE 0: LET m$=INKEY$
3310 IF m$="r" THEN  BEEP .1,1: GO TO 3700
3320 IF m$="l" THEN  BEEP .1,1: GO TO 3800
3330 IF m$="e" THEN  GO TO 6000
3340 IF m$="n" THEN  GO TO 1
3350 IF m$="v" THEN  BEEP .3,5: GO TO 7000
3360 IF m$="s" THEN  GO TO 4000
3370 IF m$="d" THEN  BEEP .3,5: GO TO 5000
3380 IF CODE m$<58 AND CODE m$>47 THEN  GO TO 5000
3390 BEEP .05,5: BEEP .1,1
3400 GO TO 3300
3700 IF y>27 THEN  GO TO 3300
3710 LET y=y+3
3720 PRINT AT x,y-3;"?"
3730 PRINT  FLASH 1;AT x,y;"*"
3740 FOR i=1 TO 50: NEXT i
3750 GO TO 3300
3800 IF y<3 THEN  GO TO 3300
3810 LET y=y-3
3820 PRINT AT x,y+3;"?"
3830 PRINT  FLASH 1;AT x,y;"*"
3840 FOR i=1 TO 50: NEXT i
```

```
3850 GO TO 3300
4000 REM Solution
4010 FOR i=1 TO 20: BEEP .05,i: BEEP .05,-4: NEXT i
4020 FOR i=1 TO 9
4030 FOR k=1 TO 4
4040 IF q$(i,k)=" " THEN  GO TO 4060
4050 PRINT AT x(i),y(i)+2*k;q$(i,k)
4060 NEXT k
4070 NEXT i
4080 FOR i=1 TO 10
4090 PRINT AT 18,3*i-1;z(i)
4100 LET g(i)=z(i)
4110 NEXT i
4120 PRINT AT 14,0; PAPER 2,,,,,,
4130 PRINT  FLASH 1;AT 15,6;"Das ist die Loesung"
4140 GO TO 6000
5000 REM Tip Eingabe
5010 LET j$=m$
5020 IF m$="d" THEN  LET j$=" ": LET g(INT (y/3)+1)=22: GO TO 5200
5030 LET g(INT (y/3)+1)=VAL j$
5040 BEEP .04,1: BEEP .04,8
5200 FOR i=1 TO 9
5210 FOR k=1 TO 4
5300 IF w$(i,k)=z$(z(y/3)+1) THEN  LET c$(i,k)=j$
5310 NEXT k
5320 NEXT i
5500 PRINT AT 18,y;j$
5600 FOR i=1 TO 9
5610 FOR k=1 TO 4
5620 PRINT AT x(i),f(i)+2*k;c$(i,k)
5630 NEXT k
5640 NEXT i
5650 PRINT AT 13,6;"????????????????????"
5800 FOR i=1 TO 10
5810 IF g(i)<>z(i) THEN  GO TO 3300
5820 NEXT i
5830 PRINT AT 14,0; PAPER 2,,,,,,
5840 PRINT  FLASH 1;AT 15,8;"G E W O N N E N"
6000 PRINT AT 21,6;"Noch ein Spiel? J/N"
6010 PAUSE 0: LET m$=INKEY$
6020 IF m$="j" THEN  GO TO 1
6030 CLS : STOP
```

```
7000 REM eine Zahl eingeben
7010 PRINT AT 18,y;z(INT (y/3+1))
7020 LET g(INT ((y/3)+1))=z(INT ((y/3)+1))
7030 FOR i=1 TO 9
7040 LET j=0: IF a(i)<1000 THEN  LET j=1
7050 IF a(i)<100 THEN  LET j=2
7060 IF a(i)<10 THEN  LET j=3
7070 FOR k=1 TO 4
7080 IF w$(i,k)=z$(z(y/3)+1) THEN  LET c$(i,k)=q$(i,k-j)
7090 PRINT AT x(i),y(i)+2*(k-j);c$(i,k)
7100 NEXT k
7110 NEXT i
7120 GO TO 3300
```



SYMBOL RATEN

Es gilt, ein mathematisches Problem zu loesen und herauszufinden, welches Symbol welcher Zahl (von 0 - 9) entspricht.
Verschiedene Symbole entsprechen verschiedenen Zahlen.
Um eine Zahl fuer ein Symbol einzugeben, wird das blinkende Sternchen (*) unter das entsprechende Symbol gefuehrt und die Zahl eingegeben.
L= * nach links, R= * nach rechts

D = loescht alten Tip
V = gibt den Wert fuer Symbol ueber dem Sternchen
S = Loesung des Problems
N = Neues Spiel
E = Ende des Spiels

Bitte eine Taste druecken!

KAPITEL 6:
S P E C T R U M   S P E K T A K U L Ä R:
T A G   U N D   N A C H T   A U F   D E R   E R D E

Entwicklung des Programms

In diesem Kapitel soll ein Programm erarbeitet werden, welches eigentlich völlig überflüssig und zu nichts Nutze ist. Zu allem Überfluß kostet es auch noch eine ganze Menge Tipparbeit, um es ordnungsgemäß zum Laufen zu bringen. Warum also diese Arbeit? Nun, wenn es fertig ist, zeigt es in eindrucksvoller Weise die grafischen Fähigkeiten des Spectrums und versorgt uns mit einer Weltkarte auf dem Bildschirm.

Um das Programm doch ein wenig interessant und abwechslungsreich zu gestalten, wollen wir noch eine Uhr einbauen und die jeweiligen Tag- und Nachtzonen auf der Erde zeigen. Diese sollen natürlich synchron mit der Uhr mitlaufen und sich auch den verschiedenen Tageslängen in Sommer und Winter automatisch anpassen.

Wenn wir das erreicht haben, ist das Programm gar nicht mehr so langweilig. Außerdem läßt sich die Weltkarte natürlich mit 'SAVE name SCREEN$' sichern und als nette Titelseite verwenden.

Sinnvollerweise werden wir die Meere der Erde in blau halten wollen. Die Tagzonen können wir gelb und die Nachtzonen schwarz gestalten. Dann müssen wir allerdings ein kleines Problem lösen. Der Spectrum besitzt zwar eine recht ansprechende hochauflösende Grafik, jedoch kann man je Printposition lediglich 2 verschiedene Farben, eine für den Hintergrund und eine für den Vordergrund, benutzen. In unserem Fall treffen aber an den 'Küsten' beim Übergang von Tag und Nacht drei verschiedene Farben aufeinander, das blaue Meer, schwarzes Land und gelbes Land.

Wenn wir diese Zonen jedoch nur jede volle Stunde auf den neuesten Stand bringen und es gerade so einrichten, daß eine Stunde auf dem Schirm der Breite einer Printpositi- on, also 8 Pixels, entspricht, können wir das Problem lösen. In diesem Fall treffen die drei Farben immer genau an der Grenze zweier Printpositionen aufeinander.

Die Daten für die Weltkarte werden wir in DATA Anweisun- gen packen und diese im Programm mit READ lesen und ver- werten. Dazu malen wir auf eine in geeigneter Größe vor- handene Weltkarte ein Raster von 102 mal 192 Quadraten . Dann suchen wir für jede Reihe von Quadraten z. B. in Süd - Nord Richtung diejenigen aus, in denen Wasser und Land zusammenkommen und geben diese in eine DATA Anweisung ein. Wenn eine Reihe fertig ist, geben wir noch zwei Nullen ein, sodaß das Programm später weiß, daß nun zur nächsten Spalte übergegangen werden muß.

Diese Prozedur sollen Sie natürlich jetzt nicht durch- führen, sie ist lediglich erklärt, damit Sie wissen, wie die DATA Anweisungen im Listing zustande gekommen sind und damit die Zeichenroutine verständlich ist.

Nachdem also diese Daten eingegeben sind (viel Spaß beim Tippen!), können wir uns an das eigentliche Programm begeben. Zunächst springen wir über die DATAs, die im Listing bei Zeile 132 beginnen, und geben ein

```
10   GO TO 8000
```

Um die Überschrift etwas herauszuheben, können wir die Großschrift - Routine aus Kapitel 4 benutzen, die wir ab Zeile 20 später eingeben. Dann geht es weiter

```
8000 CLS : LET y$="Tag und Nacht auf der Erde ": LET xpos
     =0: LET ypos= 2: GO SUB 20
8010 PRINT AT 9,0;"Waehrend das Programm laeuft,  liefer
     t        v Vorstellen                   h Halten der
      Uhr                n Normallauf
       u Korrektur der Uhr                 e Ende"
```

```
8020 PRINT'"Das Programm muss am Ersten des Monats neu ge
     startet werden"
8030 PRINT '"Falls Ende, tippen Sie bitte e  und ENTER"
8040 PRINT AT 5,0; FLASH 1;"Monat = ?(z.B. Januar = 1 usw
     )"
8050 INPUT LINE m$: IF m$="e" THEN CLS : STOP
8055 FOR i=1 TO 12: IF m$=STR$ i THEN GO TO 8065
8060 NEXT i: GO TO 8050
8065 LET mon=VAL m$
8070 PRINT AT 5,0,,AT 5,5;FLASH 1;"Uhrzeit = ?(z.B. 1445)
     "
8075 INPUT LINE m$: IF m$="e" THEN CLS : STOP
8080 IF LEN m$<>4 AND LEN m$<>3 THEN GO TO 8075
8090 FOR i=1 TO LEN m$
8100 IF m$>"9" OR m$<"0" THEN GO TO 8075
8110 NEXT i
8200 LET uhr=VAL m$: LET st=INT (uhr/100)
8210 IF st<0 OR st>24 THEN GO TO 8075
8220 LET min=uhr-100*st: IF min>59 THEN GO TO 8075
8230 PRINT AT 5,0,,: LET uhr=100*st
```

Damit haben wir relativ absturzsicher den Monat und die Uhrzeit abgefragt und die Möglichkeiten, die Uhr zu verstellen, dargelegt. Nun können wir uns an das Zeichnen der Weltkarte, die Angabe der Uhrzeit und der Zeitverschiebung und das Zeichnen der Sonne begeben. Richtig, wir werden eine Sonne zeichnen, die in roter Farbe gerade dort über der Weltkarte strahlt, wo es 12.00 Uhr ist.

```
7200 FOR i=2 TO 4: PRINT AT i,0,,: NEXT i: REM Sonne
7210 LET yy=150
7220 LET xx=224-4*uhr/50
7230 CIRCLE xx+8,yy,8
7240 INK 2
7250 FOR i=-8 TO 8
7260 FOR k=-8 TO 8
7270 IF i*i+k*k>64 THEN GO TO 7290
7280 PLOT xx+8+k,yy+i
7290 NEXT k
7300 NEXT i
```

Wenn wir nun eine kleine Routine eingeben, um die Uhrzeit auszugeben, können wir alles bis auf die Weltkarte ausgeben.

```
7000 INK 6: PRINT AT 5,0,,:REM Uhrzeit
7010 PRINT AT 5,9; FLASH 1;st;" Uhr ";min;" M E Z"
7020 RETURN
```

Jetzt folgt:

```
8300 BORDER 1: PAPER 1: INK 6: CLS
8310 GO SUB 7000: GO SUB 7200: INK 6
8320 LET y$="Tag und Nacht auf der Erde": LET xpos=0: LET
     ypos=3: INK 7: GO SUB 20: INK 6
8330 PRINT AT 7,3;"-12 -8 -4  0  +4  +8  +12"
8340 GO TO 9600
```

Das sieht schon ganz gut aus. Ab Zeile 9600 wird dann die Weltkarte gezeichnet und gegebenenfalls Uhr, Sonne und Nachtzonen auf den neuesten Stand gebracht:

```
9600 LET sek=3000: REM Ablauf
9610 GO SUB 9000
```

In der Routine ab Zeile 9000 legen wir für jeden Monat die Tag- und Nachtgebiete fest und berechnen sie für die jeweilige Uhrzeit, dann lesen wir die Landkarte ein und zeichnen sie:

```
9000 LET anf=32: RESTORE : REM paint map
9010 DIM f(12): DIM s(12)
9020 LET f(1)=3: LET f(2)=4: LET f(3)=5: LET f(4)=6: LET
     f(5)=7: LET f(6)=8
9030 LET f(7)=8: LET f(8)=7: LET f(9)=6: LET f(10)=5: LET
     f(11)=4: LET f(12)=3
9040 LET s(1)=5: LET s(2)=6: LET s(3)=7: LET s(4)=7: LET
     s(5)=8: LET s(6)=9
9050 LET s(7)=9: LET s(8)=8: LET s(9)=7: LET s(10)=6: LET
```

```
     s(11)=5: LET s(12)=5
9060 LET a1=xx-8*f(mon): LET a2=xx-8*s(mon)

9100 READ x1,x2
9110 IF x1>8000 THEN GO TO 9280
9120 IF x1=0 THEN LET anf=anf+1: GO TO 9270
9200 INK 6
9210 IF anf<a1 THEN INK 0
9220 IF anf<a2-192 THEN INK 6
9230 IF anf >=a2 THEN INK 0
9240 IF anf>=192+a1 THEN INK 6
9250 PLOT anf,x1
9260 DRAW 0,x2-x1
9270 GO TO 9100
9280 RETURN
```

Die Weltkarte ist gezeichnet und wir müssen nur noch für einen ordentlichen Ablauf sorgen, d. h. die Möglichkeit zum Uhrstellen einbauen und jede Minute die Uhr und jede Stunde das ganze Bild neu zeichnen.

```
9620 LET min=min+1
9630 LET dif=0
9640 IF min=60 THEN LET dif=1
9650 PAUSE 2325
9660 GO TO 9710
9700 PAUSE sek
9710 IF INKEY$="e" THEN CLS : STOP
9720 IF INKEY$="v" THEN LET sek=50
9730 IF INKEY$="h" THEN PAUSE 0
9740 IF INKEY$="n" THEN LET sek= 3000
9750 IF INKEY$="u" THEN GO TO 8070
```

Vorstellen und Normallauf der Uhr haben wir also dadurch geregelt, daß die Pause 'sek' den Wert ändert. Nun das Laufen der Uhr. Wir müssen etwas darauf achten, daß das Zeichnen der Weltkarte jedesmal eine gewisse Zeit benötigt und wir dann die Minutenzahl um eins erhöhen müssen, da die nächste Minute bereits begonnen hat, wenn das Bild fertig ist.

```
9760 LET min=min+1
9770 IF min>=60 THEN LET st=st+1
9780 LET uhr=100*st
9790 IF min>=60 THEN LET min=0
9800 IF st>=24 THEN LET st=0
9810 GO SUB 700
9820 IF min<>0 THEN GO TO 9700
9830 GO SUB 7200
9840 GO SUB 9000
9850 LET min=min+2+dif
9860 LET dif=0
9870 PAUSE 2325
9880 GO TO 9700
```

Nun fehlen nur noch die DATA Anweisungen. Diese sind im Gesamtlisting am Ende dieses Kapitels enthalten.

Wenn Sie einmal diese Weltkarte im Computer haben, können Sie natürlich noch weitere Möglichkeiten in das Programm einbauen. Wer möchte, kann zum Beispiel bestimmte Kontinente einzeln und in vergrößertem Maßstab zeichnen.

```
 10 GO TO 8000
 20 FOR i=1 TO LEN y$
 22 LET x=8*CODE y$(i)+15360
 24 FOR z=0 TO 7
 26 LET a=PEEK (x+z): LET b=PEEK (x+z+1)
 28 POKE USR "a"+2*z,a: POKE USR "b"+2*z+1,b
 30 NEXT z
 32 PRINT AT xpos,ypos+i-1;"A";AT xpos+1,ypos+i-1;"B"
 34 NEXT i
 36 RETURN
132 DATA 0,0
133 DATA 0,0
134 DATA 0,0
135 DATA 0,0
136 DATA 54,54,72,72,78,80,84,84,88,88,0,0
137 DATA 53,53,72,73,76,82,84,84,86,90,0,0
138 DATA 52,52,73,74,75,92,0,0
139 DATA 50,52,74,75,78,93,0,0
140 DATA 75,94,0,0
141 DATA 77,95,0,0
142 DATA 79,94,0,0
143 DATA 83,93,0,0
144 DATA 80,93,0,0
145 DATA 80,93,0,0
146 DATA 80,93,0,0
147 DATA 80,93,0,0
148 DATA 79,92,0,0
149 DATA 79,92,0,0
150 DATA 78,92,0,0
151 DATA 76,91,0,0
152 DATA 76,92,0,0
153 DATA 75,92,0,0
154 DATA 74,92,0,0
155 DATA 72,93,0,0
156 DATA 70,94,0,0
157 DATA 69,93,96,96,0,0
158 DATA 62,93,95,102,0,0
159 DATA 60,92,94,102,0,0
160 DATA 58,92,98,102,0,0
161 DATA 57,91,93,94,96,99,100,102,0,0
162 DATA 56,91,93,94,96,99,100,101,0,0
163 DATA 55,90,92,93,95,100,0,0
```

```
164 DATA 54,55,57,90,93,98,0,0
165 DATA 53,54,56,90,93,98,0,0
166 DATA 52,53,55,92,93,98,0,0
167 DATA 53,92,93,98,0,0
168 DATA 51,90,93,94,0,0
169 DATA 49,90,93,93,97,100,0,0
170 DATA 48,90,96,101,0,0
171 DATA 48,90,92,92,96,101,0,0
172 DATA 47,93,96,101,0,0
173 DATA 46,50,56,90,94,94,96,96,0,0
174 DATA 47,50,56,80,83,91,93,96,0,0
175 DATA 46,50,56,68,69,76,85,93,97,100,0,0
176 DATA 46,52,57,68,70,75,86,91,95,102,0,0
177 DATA 45,52,57,64,68,68,69,75,84,88,89,102,0,0
178 DATA 45,48,57,64,68,68,69,75,84,88,89,102,0,0
179 DATA 44,48,58,68,69,76,84,87,89,102,0,0
180 DATA 44,46,51,51,56,64,65,67,69,74,86,88,90,97,0,0
181 DATA 37,37,44,45,51,51,53,65,66,68,69,72,94,97,0,0
182 DATA 35,39,44,45,51,51,53,65,66,68,69,71,94,97,0,0
183 DATA 34,40,43,45,50,51,59,73,75,83,94,97,0,0
184 DATA 32,44,48,50,61,83,88,97,0,0
185 DATA 6,10,12,18,30,45,52,52,64,81,84,97,0,0
186 DATA 4,24,28,46,49,51,65,81,83,96,0,0
187 DATA 2,46,49,49,66,80,83,95,0,0
188 DATA 4,44,50,50,66,80,84,90,0,0
189 DATA 4,4,12,44,49,49,65,69,70,82,84,90,0,0
190 DATA 4,4,12,45,49,49,65,66,72,80,87,89,0,0
191 DATA 14,46,48,48,66,66,72,79,88,88,0,0
192 DATA 14,45,47,47,69,76,0,0
193 DATA 15,44,67,73,0,0
194 DATA 16,43,66,72,100,102,0,0
195 DATA 19,42,68,70,96,102,0,0
196 DATA 20,41,69,69,88,90,94,96,98,102,0,0
197 DATA 21,41,85,102,0,0
198 DATA 22,39,82,102,0,0
199 DATA 24,39,81,102,0,0
200 DATA 26,38,80,102,0,0
201 DATA 26,37,84,102,0,0
202 DATA 27,37,85,102,0,0
203 DATA 28,38,86,102,0,0
204 DATA 32,37,87,102,0,0
205 DATA 88,102,0,0
```

```
206 DATA 88,102,0,0
207 DATA 89,102,0,0
208 DATA 90,102,0,0
209 DATA 91,102,0,0
210 DATA 92,95,97,102,0,0
211 DATA 93,95,97,102,0,0
212 DATA 93,95,96,102,0,0
213 DATA 84,87,0,0
214 DATA 84,88,0,0
215 DATA 83,88,0,0
216 DATA 47,54,85,88,0,0
217 DATA 45,56,0,0
218 DATA 43,58,61,65,72,75,0,0
219 DATA 42,60,61,65,72,75,0,0
220 DATA 42,59,61,65,74,76,79,79,0,0
221 DATA 43,59,61,65,70,70,72,80,0,0
222 DATA 43,60,62,65,68,71,72,76,0,0
223 DATA 43,60,62,71,73,73,0,0
224 DATA 44,61,64,72,0,0
225 DATA 43,61,66,73,82,82,0,0
226 DATA 42,61,67,73,80,83,0,0
227 DATA 41,61,67,73,80,85,0,0
228 DATA 41,61,67,74,76,76,80,85,0,0
229 DATA 40,59,62,62,64,64,66,78,80,86,0,0
230 DATA 37,58,65,74,78,90,0,0
231 DATA 36,58,62,62,64,65,67,74,76,92,0,0
232 DATA 28,30,32,58,62,66,68,74,77,94,0,0
233 DATA 26,57,65,65,67,74,84,94,0,0
234 DATA 22,57,64,74,82,83,85,95,0,0
235 DATA 19,58,62,78,81,83,85,96,0,0
236 DATA 18,58,61,78,81,84,86,96,0,0
237 DATA 19,57,59,59,62,86,88,97,0,0
238 DATA 20,58,64,80,82,97,0,0
239 DATA 21,58,60,64,66,94,0,0
240 DATA 23,57,60,64,66,94,0,0
241 DATA 25,52,54,58,60,86,88,94,0,0
242 DATA 25,51,54,64,67,85,88,93,0,0
243 DATA 28,50,53,64,67,87,89,91,0,0
244 DATA 29,48,53,63,66,84,88,90,0,0
245 DATA 30,34,38,47,52,63,65,88,0,0
246 DATA 30,33,39,46,51,88,0,0
247 DATA 39,45,47,91,0,0
```

```
248 DATA 25,31,40,45,47,88,0,0
249 DATA 25,32,44,46,48,62,68,90,0,0
250 DATA 48,51,54,60,67,89,93,101,0,0
251 DATA 48,52,56,89,96,100,0,0
252 DATA 49,53,56,90,96,96,100,100,0,0
253 DATA 50,52,56,89,91,92,0,0
254 DATA 50,51,55,90,0,0
255 DATA 51,51,55,90,0,0
256 DATA 55,90,0,0
257 DATA 55,90,0,0
258 DATA 54,92,94,96,0,0
259 DATA 53,97,0,0
260 DATA 52,99,0,0
261 DATA 51,87,89,100,0,0
262 DATA 50,87,89,100,0,0
263 DATA 50,88,0,0
264 DATA 48,88,0,0
265 DATA 46,90,92,99,0,0
266 DATA 44,90,92,97,0,0
267 DATA 43,44,46,99,0,0
268 DATA 49,100,0,0
269 DATA 50,100,0,0
270 DATA 51,100,0,0
271 DATA 52,100,0,0
272 DATA 53,100,0,0
273 DATA 42,42,52,100,0,0
274 DATA 41,42,51,100,0,0
275 DATA 40,42,51,100,0,0
276 DATA 39,41,43,100,0,0
277 DATA 38,40,42,43,45,100,0,0
278 DATA 37,39,40,42,44,100,0,0
279 DATA 36,37,39,41,43,100,0,0
280 DATA 35,35,37,38,42,100,0,0
281 DATA 34,35,41,100,0,0
282 DATA 34,35,37,39,44,44,48,100,0,0
283 DATA 24,25,34,35,37,40,49,100,0,0
284 DATA 23,26,34,35,37,41,49,100,0,0
285 DATA 19,26,34,34,38,42,50,99,0,0
286 DATA 19,27,41,41,43,43,46,46,53,99,0,0
287 DATA 20,28,34,34,36,40,44,44,54,55,57,57,60,60,64,98,0,0
288 DATA 21,30,36,40,44,45,48,48,63,99,0,0
289 DATA 21,31,44,45,64,99,0,0
```

```
 290 DATA 22,32,61,100,0,0
 291 DATA 22,32,61,100,0,0
 292 DATA 22,33,59,59,62,62,64,97,0,0
 293 DATA 22,33,58,60,66,97,0,0
 294 DATA 21,33,39,40,59,60,66,96,0,0
 295 DATA 19,32,37,38,60,61,66,96,0,0
 296 DATA 18,31,34,38,61,63,66,74,76,98,0,0
 297 DATA 17,31,34,38,62,64,66,66,80,98,0,0
 298 DATA 16,33,35,38,65,74,80,98,0,0
 299 DATA 16,33,36,38,65,67,72,72,80,97,0,0
 300 DATA 13,15,17,31,35,36,66,66,80,97,0,0
 301 DATA 13,15,17,30,35,36,80,97,0,0
 302 DATA 19,28,35,35,80,96,0,0
 303 DATA 23,25,80,94,0,0
 304 DATA 81,94,0,0
 305 DATA 73,79,82,94,0,0
 306 DATA 71,80,82,94,0,0
 307 DATA 72,80,82,94,0,0
 308 DATA 74,80,82,93,0,0
 309 DATA 74,81,82,91,0,0
 310 DATA 76,81,83,91,0,0
 311 DATA 78,91,0,0
 312 DATA 13,13,80,94,0,0
 313 DATA 12,14,80,92,0,0
 314 DATA 13,15,81,94,0,0
 315 DATA 14,15,17,19,82,94,0,0
 316 DATA 15,18,83,94,0,0
 317 DATA 15,17,84,94,0,0
 318 DATA 16,17,84,93,0,0
 319 DATA 16,16,83,83,86,92,0,0
 320 DATA 88,92,0,0
 321 DATA 87,91,0,0
 322 DATA 86,90,0,0
 323 DATA 90,90,0,0
 500 DATA 9999,9999
7000 INK 6: PRINT AT 5,0,,: REM uhrzeit
7010 PRINT AT 5,9; FLASH 1;st;" Uhr ";min;"  M E Z"
7020 RETURN
7200 FOR i=2 TO 4: PRINT AT i,0,,: NEXT i: REM sonne
7210 LET yy=150
7220 LET xx=224-4*uhr/50
7230 CIRCLE xx+8,yy,8
```

```
7240 INK 2
7250 FOR i=-8 TO 8
7260 FOR k=-8 TO 8
7270 IF i*i+k*k>64 THEN  GO TO 7290
7280 PLOT xx+8+k,yy+i
7290 NEXT k
7300 NEXT i
7310 RETURN
8000 CLS : LET y$="Tag und Nacht auf der Erde ? pcb": LET xpos=0: LET ypos=0: GO
 SUB 20
8010 PRINT AT 9,0;"Waehrend das Programm laeuft,   liefert      v Vorstellen
                h Halten der Uhr                  n Normallauf
u Korrektur der Uhr              e Ende"
8020 PRINT '"Das Programm muss am ersten des Monats neu gestartet werden"
8030 PRINT '"Falls Ende, tippen Sie bitte e  und ENTER"
8040 PRINT AT 5,0; FLASH 1;"Monat = ?(z.B. Januar = 1 usw)"
8050 INPUT  LINE m$: IF m$="e" THEN  CLS : STOP
8055 FOR i=1 TO 12: IF m$=STR$ i THEN  GO TO 8065
8060 NEXT i: GO TO 8050
8065 LET mon=VAL m$
8070 PRINT AT 5,0,,AT 5,5; FLASH 1;"Uhrzeit = ?(z.B. 1445)"
8075 INPUT  LINE m$: IF m$="e" THEN  CLS : STOP
8080 IF LEN m$<>4 AND LEN m$<>3 THEN  GO TO 8075
8090 FOR i=1 TO LEN m$
8100 IF m$(i)>"9" OR m$(i)<"0" THEN  GO TO 8075
8110 NEXT i
8200 LET uhr=VAL m$: LET st=INT (uhr/100)
8210 IF st<0 OR st>24 THEN  GO TO 8075
8220 LET min=uhr-100*st: IF min>59 OR min<0 THEN  GO TO 8075
8230 PRINT AT 5,0,,: LET uhr=100*st
8300 BORDER 1: PAPER 1: INK 6: CLS
8310 GO SUB 7000: GO SUB 7200: INK 6
8320 LET y$="Tag und Nacht auf der Erde": LET xpos=0: LET ypos=3: INK 7: GO SUB
20: INK 6
8330 PRINT AT 7,3;"-12 -8  -4   0  +4  +8 +12"
8340 GO TO 9600
9000 LET anf=32: RESTORE : REM paint map
9010 DIM f(12): DIM s(12)
9020 LET f(1)=3: LET f(2)=4: LET f(3)=5: LET f(4)=6: LET f(5)=7: LET f(6)=8
9030 LET f(7)=8: LET f(8)=7: LET f(9)=6: LET f(10)=5: LET f(11)=4: LET f(12)=3
9040 LET s(1)=5: LET s(2)=6: LET s(3)=7: LET s(4)=7: LET s(5)=8: LET s(6)=9
9050 LET s(7)=9: LET s(8)=8: LET s(9)=7: LET s(10)=6: LET s(11)=5: LET s(12)=5
```

```
9060 LET a1=xx-8*f(mon): LET a2=xx+8*s(mon)
9100 READ x1,x2
9110 IF x1>8000 THEN  GO TO 9280
9120 IF x1=0 THEN  LET anf=anf+1: GO TO 9270
9200 INK 6
9210 IF anf<a1 THEN  INK 0
9220 IF anf<a2-192 THEN  INK 6
9230 IF anf>=a2 THEN  INK 0
9240 IF anf>=192+a1 THEN  INK 6
9250 PLOT anf,x1
9260 DRAW 0,x2-x1
9270 GO TO 9100
9280 RETURN
9600 LET sek=3000: REM ablauf
9610 GO SUB 9000
9620 LET min=min+1
9630 LET dif=0
9640 IF min=60 THEN  LET dif=1
9650 PAUSE 2325
9660 GO TO 9710
9700 PAUSE sek
9710 IF INKEY$="e" THEN  CLS : STOP
9720 IF INKEY$="v" THEN  LET sek=50
9730 IF INKEY$="h" THEN  PAUSE 0
9740 IF INKEY$="n" THEN  LET sek=3000
9750 IF INKEY$="u" THEN  GO TO 8070
9760 LET min=min+1
9770 IF min>=60 THEN  LET st=st+1
9780 LET uhr=100*st
9790 IF min>=60 THEN  LET min=0
9800 IF st>=24 THEN  LET st=0
9810 GO SUB 7000
9820 IF min<>0 THEN  GO TO 9700
9830 GO SUB 7200
9840 GO SUB 9000
9850 LET min=min+2+dif
9860 LET dif=0
9870 PAUSE 2325
9880 GO TO 9700
```

KAPITEL 7:
Z U S A T Z G E R Ä T E U N D E R W E I T E R U N G E N

Einleitung

Früher oder später passiert es fast jedem Besitzer eines Heimcomputers: man schaut sich um, was man denn so als Zubehör erwerben könnte, um das System zu verbessern, auszubauen oder einfach seinem Spieltrieb noch besser frönen zu können.

Von der Firma Sinclair selbst werden eine Reihe von Zusatzgeräten vertrieben, wie das Interface 1 und die Microdrives, welche die Möglichkeiten des Computers ungemein erweitern. Dank der großen Verbreitung des Spectrums hat sich aber auch eine große Anzahl von unabhängigen Firmen auf die Herstellung von Peripherie für Sinclair Computer spezialisiert. So hat man für den Spectrum eine umfangreiche Palette wie für kaum ein zweites Gerät zur Auswahl und dem Anwender sind fast nur durch den eigenen Geldbeutel Grenzen gesetzt.

Als erstes kommt meist der Wunsch nach einem Drucker auf, da gerade das Editieren langer Programme auf die Dauer mühsehlig ist, wenn man nur den Bildschirm zur Verfügung hat. Das preiswerteste Angebot hier ist der Minidrucker von Sinclair. Er ist direkt an den Spectrum anschliessbar und schreibt auf Metallpapier. Dies Papier ist allerdings nicht ganz billig. Auch ist die Druckqualität nicht gerade überwältigend. Mittlerweile ist die Produktion dieses Gerätes eingestellt worden. Einige Firmen stellen aber ebenfalls kleine Drucker her, die darüberhinaus auf normalem Thermopapier oder sogar Normalpapier schreiben und einen robusteren Eindruck hinterlassen.

Benötigt man den Drucker nur für gelegentliche Listings,

erscheint der Alphacom Drucker ausreichend, anspruchsvollere Benutzer können aber noch etwas zulegen und mit dem Seikosha GP 50S einen echten Matrix-Drucker erwerben. Mit einem Centronics - Parallelinterface lassen sich aber auch eine Vielzahl der eigentlich für größere Computer hergestellten Drucker mit 80 Zeichen pro Zeile und mehr anschliessen. Diese haben zum Teil ein sehr gutes Schriftbild und der Spectrum läßt sich dann durchaus als Grundbaustein eines Textverarbeitungssystems benutzen.

Möchte man tatsächlich viel Text mit dem Spectrum schreiben, so wird man nach einiger Zeit nicht mehr mit den Gummitasten zufrieden sein. (Das gilt natürlich nicht für die Spectrum Plus Besitzer.) Diese Marktlücke ist von vielen Firmen erkannt worden und so gibt es auch hier reichlich Abhilfe. Sinclair hat es den Herstellern von Zusatztastaturen aber auch recht leicht gemacht. Die Spectrum - Tastatur ist nämlich ein separater Teil des Computers. Schraubt man den Spectrum auf, so sieht man, daß von der Tastatur aus zwei Flachbandkabel zur Platine führen, die lediglich in Stecker eingesteckt sind. Eine neue Tastatur wird daher meist so angeschlossen, daß die beiden Kabel herausgezogen werden und entsprechend die beiden Kabel der neuen Tastatur eingesteckt werden.

Hat man die etwa 200 DM für eine neue Tastatur eingesetzt und den Eingriff in den Computer gewagt, wird man allerdings wirklich belohnt. Dies ganze Buch ist mit einer solchen Konfiguration geschrieben worden.

Das Interface 1

Das vielseitigste Zusatzgerät zum Spectrum ist ohne Frage das Interface 1 von Sinclair Research. Zu einem sehr günstigen Preis bietet es die Möglichkeit zum Zugriff auf schnelle Massenspeicher, den Microdrives, als auch eine eingebaute RS 232 Schnittstelle, über die sich Verbindung zu anderen Computern, Druckern und auch Modems herstellen läßt. Weiterhin erlaubt es, ein lokales Netzwerk zwischen bis zu 64 Spectrums (und Sinclair QLs) aufzubauen. Dabei bleibt die Möglichkeit zum Anschluß des Tonbands und des ZX - Druckers erhalten.

Das Interface 1 beinhaltet auch ein zusätzliches ROM von 8 KByte Umfang, welches sich zu dem 16 KByte ROM des Spectrums gesellt. Dieses zusätzliche ROM erweitert den Befehlssatz des Spectrums um die Anweisungen

CAT
CLOSE
ERASE
FORMAT
MOVE
OPEN

und erweitert die Möglichkeiten der Anweisungen

CLS
CLEAR
INPUT
INKEY$
PRINT
LPRINT
LIST
LLIST
SAVE
LOAD
MERGE
VERIFY

Die Bedeutung dieser Anweisungen sind im Anhang gegeben. Hier sollen einige Besonderheiten herausgestellt werden.

Die Anweisung 'CAT' dient dazu, ein Inhaltsverzeichnis einer Microdrivekassette zu erhalten. So liefert der Befehl 'CAT 1' den Namen der Microdrivekassette in Microdrive 1 und die Namen der ersten 50 nicht geschützten Files, die darauf enthalten sind. Ein geschützter File ist einer, dessen Name mit CHR$ 0 beginnt. Solche Files werden von 'CAT' ignoriert. Daneben wird noch der noch verfügbare Speicherplatz in KBytes angezeigt.

Diese Liste wird bei der obigen Form der Anweisung auf den Bildschirm ausgegeben. Möchte man diese Liste zum Beispiel auf den Drucker geschrieben haben, muß der entsprechende Strom geöffnet sein und die Form

```
CAT #n;1
```

eingegeben werden. Dabei ist 'n' die Nummer des entsprechenden Stroms.

Benutzt man die FORMAT Anweisung, um die Übertragungsrate über die RS 232 Schnittstelle festzulegen, so dürfen die Werte 50, 110, 300, 600, 1200, 2400, 4800, 9600 und 19200 eingegeben werden. Wird kein Wert eingegeben, so beträgt die Baud Rate 9600. Gibt man eine nicht vorgesehene Zahl ein, so wird immer die nächst kleinere 'erlaubte' Baud Rate genommen. Das Minimum ist allerdings immer 50.

CLs# ist mit dem Interface ebenfalls erlaubt. Im Gegensatz zur einfachen CLS Anweisung wird nicht nur der Display-File gelöscht, sondern weiterhin gesetzt:

INK	wird 0, also schwarz
PAPER	wird 7, also weiss
BORDER	wird ebenfalls 7
INVERSE	wird auf 0 gesetzt

BRIGHT	wird auf 0 gesetzt
OVER	wird auf 0 gesetzt
FLASH	wird auf 0 gesetzt

Die Anweisung CLEAR# löscht eventuell gesetzte Daten über Ströme. Alle Ströme werden geschlossen und Daten, die sich vielleicht in irgendwelchen Zwischenspeichern befinden, gehen im Gegensatz zur Anweisung CLOSE# verloren.

Bis zu acht Microdrive - Laufwerke können an das Interface 1 gleichzeitig angeschlossen und über dies gesteuert werden. Mit diesen Laufwerken können spezielle Endlos - Kassetten beschrieben und gelesen und damit als Speicher genutzt werden.

Jede dieser Kassetten hat ein etwa 5 m langes Band, welches für einen Durchlauf etwa 7 Sekunden benötigt. Im Mittel beträgt damit die Zugriffszeit auf Daten 3.5 Sekunden. Jede Kassette kann etwa 90 KByte Daten speichern. Es empfiehlt sich, das Formatieren einer Kassette mehrmals durchzuführen, da der verfügbare Speicherplatz bei solchen Versuchen variieren kann.

Eine Besonderheit des Interface 1 ist das Lokale Netzwerk LAN, welches die direkte Verbindung von bis zu 64 Spectrums und/oder Sinclair QL's erlaubt. Diese Verbindung wird mit einem einfachen zweiadrigen Kabel durchgeführt. Daten lassen sich mit einer Geschwindigkeit von 5 KByte pro Sekunde austauschen.

Mit der Anweisung FORMAT "n",7 wird einem Spectrum zum Beispiel die Stationsnummer 7 zugeordnet. Diese Stationsnummer muß zwischen 1 und 64 liegen. Den gleichen Effekt erreicht man übrigens mit dem Befehl POKE 23749,7.

Der Datenaustausch ist denkbar einfach, es werden die Befehle LOAD, SAVE, MERGE usw. benutzt, wobei anstelle des "m" für Microdrives ein "n" für Netzwerk tritt. Die Einsatzmöglichkeiten sind vielfältig. Wie das Handbuch

zum Interface 1 beschreibt, können zum Beispiel mehrere Spectrum Besitzer ein teures peripheres Gerät wie einen Drucker gemeinsam benutzen.

Es lassen sich völlig neue Spiele konzipieren, wenn mehrere Spectrums miteinander kommunizieren können. Auch können Programme erstellt werden, bei denen zwei Rechner verschiedene Teile eines Programms bearbeiten. Die Ergebnisse der Rechnungen können ausgetauscht werden und auf zwei Bildschirmen (oder bei mehreren Rechnern auch mehr) dargestellt werden. Sowohl auf der Hardware - als auch auf der Softwareseite liegt hier ein interessantes Betätigungsfeld offen.

Serielle Drucker

Besitzern des Interface 1 und damit der RS 232 Schnittstelle steht neben der Verwendung von reinen Druckern auch die Möglichkeit offen, eine Schreibmaschine mit einer entsprechenden Schnittstelle als Drucker zu verwenden. Unter den vielen Modellen sei hier das Modell EP44 der Firma Brother herausgehoben. Der Grund hierfür liegt in der Vielseitigkeit dieses Gerätes.

Diese Schreibmaschine schreibt entweder ohne Schreibband auf Thermopapier (was nicht zu billig ist) oder mit einem Schreibband (was auch nicht billig ist) ist, auf Normalpapier. Das Schriftbild ist Dank einer 18 x 24 Punktmatrix sehr gut. Die Verwendung als Drucker in Verbindung mit dem Spectrum kann kaum einfacher verwirklicht werden. Man steckt das RS 232 Kabel in den Stecker und gibt als Parameter für die Datenübertragung die folgenden Werte ein:

BAUD RATE	beliebig, jedoch muß diese natürlich mit dem Wert des Spectrum übereinstimmen
BIT LENGTH	8
PARITY	N
NEW LINE	CR + LF
CODE	7 BIT
ER	Y

Dann wird der entsprechende Schalter auf 'TERMINAL' gestellt und in den Spectrum die Anweisung

```
OPEN#3,"t": FORMAT "t";baud rate
```

eingegeben, wobei 'baud rate' für die entsprechende Zahl steht, die bei der EP 44 eingegeben wurde.

Von nun an wirken LPRINT und LLIST Anweisungen auf die EP 44. Das liegt natürlich daran, daß LPRINT und LLIST nichts anderes als PRINT#3 und LIST#3 bedeuten. COPY kann leider nicht klappen, da die EP 44 nicht grafikfähig ist.

Die Schreibmaschine verfügt über eine ganze Reihe von Fähigkeiten wie Textspeicherung und Datenfernübertragung. Ja, sogar eine Art Texteditor ist 'eingebaut'. Was allerdings für Spectrumbesitzer ohne Zusatztastatur besonders interessant ist, ist die Möglichkeit, die EP 44 als Eingabegerät für den Spectrum zu benutzen.

Eine einfache Routine, mit der dieses bewerkstelligt werden kann, ist die folgende:

```
10   OPEN#3,"t": FORMAT "t";300
20   BORDER 0
100  LET m$=INKEY#3
110  IF CODE m$>31 THEN PRINT m$;
150  GO TO 100
```

Alles was nun auf der EP 44 eingegeben wird, erscheint auf dem Bildschirm. Diese kleine Routine verdeutlicht das Prinzip, mit dem man in seinen Programmen die Eingabe über die Tastatur der Schreibmaschine machen kann. Das ist aber eine gute Sache, man kann die EP 44 sowohl als Drucker als auch in beschränktem Maße als Tastatur benutzen, was für viele Spectrum Besitzer eine Ersparnis von rund 200 DM für eine separate Tastatur bedeutet.

Möchte man vom Spectrum aus gleich auf die Schreibmaschine schreiben und vielleicht einen Dialogbetrieb zwischen den beiden unterhalten, braucht man in die obige Routine nur die beiden Zeilen

```
120  LET n$=INKEY$
```

```
130  IF CODE n$>31 THEN PRINT#3;n$
```

einzufügen. Natürlich läßt sich in der EP 44 gespeicherter Text so auf den Bildschirm bringen. Die Möglichkeiten sind hier recht vielfältig.

Lightpen

Ein zumindest sehr nettes Zusatzgerät zum Spectrum (wie auch zu jedem anderen Computer) ist ein Lichtgriffel, der auf gut deutsch oft auch Lightpen genannt wird. Mit diesem Stift lassen sich Dinge anstellen, die für einen Laien wie Zauberei anmuten. Es lassen sich Linien, Kreise und andere Figuren auf den Bildschirm 'zeichnen'. Spektakulär ist auch die Benutzung des Stifts bei der Auswahl aus einem Menü.

Allerdings darf man von dem Stift nicht wirkliche Wunder erwarten. Wer zum Beispiel den Aachener Dom auf den Schirm zeichnen möchte und dabei gerade Linien gerade sein lassen möchte, braucht ziemlich lange Übung!

Wie funktioniert ein Lichtgriffel? Im Prinzip ist er ein Lichtempfänger, der dieses in ein für den Computer verständliches elektrisches Signal umwandelt. Das Fernsehbild wird mit einem Elektronenstrahl jede Sekunde 25 mal geschrieben, indem dieser Strahl in 625 waagerechten Linien über den Schirm streicht. Hält man nun den Lichtgriffel gegen den Fernsehschirm, so 'sieht' er immer dann Licht, wenn der Elektronenstrahl gerade auf seine Position gerichtet ist. Eben dann schickt der Lichtgriffel ein Signal zum Computer.

Dieser braucht dann nur noch auszurechnen, wieviel Zeit vergangen ist, seit der Strahl mit dem Bild begonnen hat, also an der linken oberen Ecke war. Aus dieser Zeitdifferenz kann er dann exakt bestimmen, an welcher Position der Lichtgriffel war.

Wenn auch komplizierte Gemälde kaum mit dem Griffel freihändig angefertigt werden können, läßt sich mit geeigneter Software eine Menge erreichen. Auch ist der Lichtgriffel ein angenehmes Eingabegerät für den Computerlaien, da er einfach auf die Punkte zeigen muß, die er erreichen will.

Möchte man lediglich die Auswahl aus einem Menü mit einem Lichtgriffel durchführen, ist dies besonders einfach. Jedoch muß aufgepaßt werden, daß der Stift nicht schon auf dem Weg zum Schirm Licht empfängt, oder während der Benutzer über den Schirm streicht, um am gewünschten Punkt anzuhalten.

Eine einfache Methode, solche 'Fehler' auszuschalten, besteht darin, zweimal die Abfrage auf die Zeile zu machen, und zwischen diesen Abfragen eine kleine Pause verstreichen zu lassen. Wenn beide Abfragen das gleiche Ergebnis geliefert haben, ist der Stift offensichtlich ruhig an einer Stelle gehalten worden, um eine Auswahl zu treffen.

Das kann in einem Programm etwa so aussehen:

```
100  LET zeile1=USR 63109 (oder wo immer die
                          Routine vom Hersteller
                          plaziert wurde)
110  PAUSE 10
120  LET zeile2=USR 63109
130  IF zeile1=zeile2 THEN GO TO .....
140  GO TO 100
```

Die Pause in Zeile 110 kann ganz nach Belieben verlängert oder verkürzt werden, bis ein optimales Ergebnis erzielt wird.

Textverarbeitung

Das wahrscheinlich am weitesten verbreitete Textverarbeitungsprogramm für den Spectrum ist TASWORD II. Dies aus gutem Grund, denn das Programm ist schlicht gesagt sehr gut. Darüberhinaus ist es auch sehr benutzerfreundlich. Doch zunächst einige Bemerkungen zum Programm selbst.

Es wird mit einer ausführlichen deutschen Beschreibung geliefert, diese benötigt man jedoch nicht sehr lange, da die Handhabung sehr einfach ist. Darüberhinaus lassen sich die wesentlichen Merkmale mit Hilfe zweier 'Help - Seiten' auch während der Arbeit an einem Dokument auf den Bildschirm holen.

Das Programm eröffnet alle Möglichkeiten, die man von einer Textverarbeitung erwartet. Mit dem Kursor kann in allen vier Richtungen herumgefahren werden. Zu jeder Zeit kann damit an jeder Stelle korrigiert, eingefügt oder gelöscht werden. Das Programm zeigt im Normalfall den Text mit 64 Zeichen pro Zeile. Wem dies zu klein ist, der kann seinen Text natürlich auch in der normalen Spectrum - Schrift bearbeiten.

Der Text wird bei der Eingabe automatisch formatiert, das heißt, daß in jede Zeile soviele Leerstellen eingefügt werden, daß ein gerader Rand an der rechten Seite entsteht. Selbst nach dem Einfügen oder Löschen von Text kann dieser Randausgleich jederzeit wiederhergestellt werden.

Worte lassen sich zentrieren und nach links oder rechts schieben. Zur Korrektur lassen sich Worte im Text suchen und ersetzen, ja auch ganze Blöcke von Text können verschoben oder kopiert werden. Auch läßt sich bereits gespeicherter Text mit einem anderen Dokument 'MERGEn'.

Die oben angesprochene Benutzerfreundlichkeit kommt darin

zum Ausdruck, daß das Programm zum eigenen Bedarf kopiert werden kann. Es ist sogar im Menü ein Punkt vorgesehen, der dieses automatisch durchführt. Dies hat zur Folge, daß man das Programm gemäß der Anleitung an seinen eigenen Drucker anpassen, und diese Version dann als Arbeitsversion abspeichern und nutzen kann.

Ein weiterer Vorteil ist auch, daß der Steuerteil des Programms in Basic geschrieben ist und sich damit zur Benutzung mit Interface 1 und Microdrives sehr leicht umschreiben läßt. Wer hat sich nicht schon geärgert, wenn er ein schönes Schachprogramm hat und dies nach Anschaffung der Microdrives überspielen will, es aber trotz aller Anstrengungen nicht gelingen will und man immer noch Minuten auf das Laden warten muß.

Schreibt man Tasword II auf Microdrives um, muß man etwas sparsam mit dem Speicherplatz umgehen, da für den Basicteil nur die Speicherplätze bis 31999 zur Verfügung stehen. Es empfiehlt sich daher, möglichst viele platzsparende Maßnahmen durchzuführen. Im folgenden ist ein Beispiel angegeben, wie das Programm bei Benutzung zweier Microdrives und Kassettentonband zur Speicherung und einem Drucker über die RS 232 Schnittstelle funktioniert. Auch sind die Farben des Bildschirms verändert, was natürlich Geschmacksache ist.

```
   5 BORDER 7: PAPER 7: INK 0: LET q$="pcb"
  10 CLS : LET a=USR 64330: GO TO 20
  15 BRIGHT 1: INK 0: PAPER 7: BORDER 7: POKE 23609,2: CLEAR 31999: LET q$="pcb"
: GO SUB 4000: LOAD *"m";1;"taswordmc"CODE : CLS : LET a=USR 59081: GO TO 10
  20 BORDER NOT PI: PAPER NOT PI: INK VAL "6": CLS : LET a=64*INT (a/64+0.99): I
F a=NOT PI THEN  GO TO 3000
  25 LET a$=" text file"
  26 GO SUB 4000: LET v=VAL "31":  PRINT AT VAL "4",NOT PI;"print";a$;TAB v;"p"
  28  PRINT '"save";a$;TAB v;"s"
  30 PRINT '"load";a$;TAB v;"j"
  35 PRINT '"merge";a$;TAB v;"m"
  40 PRINT '"return to";a$;TAB v;"y"
  45  PRINT '"define graphics/printer";TAB v;"g"
  50 PRINT '"save tasword";TAB v;"t"
  55 PRINT '"into Basic";TAB v;"b"
  70 PRINT AT 20,VAL "7";"press key"
  80 LET a$=INKEY$: IF a$="" THEN  GO TO 80
  90 LET i=0: LET b=CODE a$: IF b<97 THEN  LET b=b+32
 110 IF b=115 THEN  LET i=VAL "6"
 120 IF b=106 THEN  LET i=VAL "8"
 125 IF b=116 THEN  LET i=16
 130 IF b=112 THEN  LET i=VAL "4"
 140 IF b=121 THEN  LET i=12
 150 IF b=109 THEN  LET i=10
 160 IF b=103 THEN  LET i=14
 170 IF b=98 THEN  LET i=18
 180 IF i>VAL "0" THEN  PRINT AT i,31; FLASH VAL "1";CHR$ b;: GO TO 500
 190 GO TO 80
 200 CLS : GO SUB 4000: PRINT AT VAL "4",VAL "8";"PRINT OPTIONS": PRINT "  just
press ENTER for default    values given in brackets"
 210 LET i=VAL "8": LET j0=23: PRINT AT i,NOT PI;"Line spacing? (1)": GO SUB 600
0: IF a$="" THEN  LET a$="1"
 215 POKE 62235,VAL a$
 220 LET i=10: PRINT AT i,NOT PI;"Start at line? (1)": GO SUB 6000: IF a$="" THE
N  LET a$="1"
 230 LET c=64*(INT VAL a$-1): LET b=c+FN p(62216): LET x=60045: GO SUB 950
 240 LET i=12: PRINT AT i,NOT PI;"Finish at line? (last)": GO SUB 6000: IF a$=""
 THEN  LET b=a-c: GO TO 250
 245 LET b=64*INT VAL a$-c
 250 RANDOMIZE USR 59806: RANDOMIZE USR (FN p(62472))
 260 CLS : PRINT AT 20,NOT PI;"Press the q key to quit printing"
 265 FORMAT "t";300: OPEN  3;"t"
```

```
 270 LET x=60049: GO SUB 950
 275 LET c=PEEK 62470: IF c<>0 THEN  LPRINT CHR$ c
 280 RANDOMIZE USR 60038
 285 LET c=PEEK 62471: IF c<>0 THEN  LPRINT CHR$ c
 290 RANDOMIZE USR 59806: CLOSE  3: GO TO 5
 300 CLS : GO SUB 4000: PRINT "Printer control graphics chars:"
 305 LET b=VAL "4": FOR i=NOT PI TO 15: LET b=ABS (b-4): PRINT AT i+4,b;i+128;CH
R$ (i+128)
 307 FOR j=NOT PI TO VAL "3": LET c=PEEK (60860+i*4+j): LET a$=STR$ c: IF c=255
THEN  LET a$=""
 308 PRINT AT i+4,10+4*j;a$: NEXT j: NEXT i
 320 INPUT "Type graphics character code 128-143 (ENTER if finished)";a$
 325 IF a$="" THEN  GO TO 400
 340 LET b=VAL a$: IF b<128 OR b>143 THEN  GO TO 320
 350 PRINT AT 21,VAL "3"; FLASH VAL "1";b; FLASH NOT PI;" ";CHR$ b
 355  FOR j=NOT PI TO VAL "3": POKE (60348+b*4+j),255: NEXT j
 360 FOR j=NOT PI TO VAL "3": INPUT "Code? (ENTER if finished)";a$: IF a$="" THE
N  GO TO 300
 370 POKE (60348+b*4+j),VAL a$: PRINT AT 21,10+4*j;VAL a$: NEXT j: GO TO 300
 400 CLS : GO SUB 4000: PRINT AT VAL "3",NOT PI;"Reset interface/printer codes?"
: LET i=VAL "5": GO SUB 920: CLS : IF i=NOT PI THEN  GO TO 25
 401 GO SUB 4000: PRINT AT 3,0;"just ENTER to keep values given:"
 403 LET j0=27: LET i=8: LET j=0: LET a$="Interface control code1=": LET x=60924
: GO SUB 850
 404 LET i=9: LET j=18: LET a$="code2=": LET x=62470: GO SUB 850
 405 LET i=10: LET j=18: LET a$="code3=": LET x=62471: GO SUB 850
 406 LET i=11: PRINT AT i,18;"code4=";FN p(62472): LET i=12: LET j0=24: GO SUB 6
000: LET j0=27: IF a$<>"" THEN  LET b=VAL a$: LET x=62472: GO SUB 950
 410 LET i=14: LET j=0: LET a$="Printer carriage return=": LET x=60925: GO SUB 8
50
 420 LET i=16: LET j=0: LET a$="Printer linefeed=": LET x=60926: GO SUB 850
 430 LET i=18: LET j=0: LET a$="Left margin on printing=": LET x=60927: GO SUB 8
50
 490 GO TO 20
 500 PRINT AT 20,0;" press the "; FLASH 1;"ENTER"; FLASH 0;" key to proceed";AT
21,0;"  press "; FLASH 1;"c"; FLASH 0;" to change the choice  "
 510 LET a$=INKEY$: IF a$="c" THEN  GO TO 20
 520 IF CODE a$=13 THEN  GO TO 600
 530 GO TO 510
 600 IF b=116 THEN  GO TO 700
 610 IF b=121 THEN  CLS : GO TO 5
 620 IF b=115 THEN  CLS : GO TO 1000
```

```
 630 IF b=109 THEN  GO TO 2000
 640 IF b=106 THEN  LET a=USR 59081: LET a=0: GO TO 2000
 650 IF b=112 THEN  GO TO 200
 660 IF b=103 THEN  GO TO 300
 699 CLS : STOP
 700 CLS : LET i=8: LET a$="tasword": SAVE *"m";1;a$ LINE 15
 710 SAVE *"m";1;a$+"mc"CODE 54784,10751
 770 PRINT AT 19,0;
 780 VERIFY *"m";1;a$: PRINT AT 21,0;"tasword: basic O.K.;";AT 19,0;
 790 VERIFY *"m";1;a$+"mc"CODE 54784,10751: PRINT AT 21,20;" m/code O.K.": GO TO
 25
 850 PRINT AT i,j;a$;PEEK x: GO SUB 6000: IF a$<>"" THEN  POKE x,VAL a$
 860 RETURN
 920 PRINT AT i,4;"press  y  for yes";AT i+2,11;"n  for no"
 930 IF INKEY$="n" OR INKEY$="N" THEN  LET i=0: RETURN
 940 IF INKEY$="y" OR INKEY$="Y" THEN  LET i=1: RETURN
 945 GO TO 930
 950 POKE x,b-256*INT (b/256): POKE (x+1),INT (b/256): RETURN
1000 LET b=FN p(62216): CLS
1005 PRINT AT VAL "8",NOT PI;"Name of text file for saving?": LET i=10: LET j0=N
OT PI: GO SUB 6000
1010 IF a$="" OR LEN a$>10 THEN  GO TO 1005
1020 INPUT "MD (1 or 2) or c for Cassette"; LINE m$: IF m$<>"1" AND m$<>"2" AND
m$<>"c" THEN  GO TO 1020
1022 IF m$="c" THEN  GO TO 1035
1025 ERASE "m";VAL m$;a$
1030 LET i=12: SAVE *"m";VAL m$;a$CODE b,a: LET i=14: VERIFY *"m";VAL m$;a$CODE
b,a: GO TO 1040
1035 LET i=12: SAVE a$CODE b,a: LET i=14: PRINT "rewind to verify and press key"
: PAUSE 0: VERIFY a$CODE b,a
1040 CLS : PRINT AT 8,NOT PI;"text file ";a$;" saved:";AT 10,NOT PI;a;" bytes,",
a/PEEK 62237;" lines"
1100 PRINT AT 21,6;"text file verified": PAUSE 150: GO TO 25
2000 CLS : BORDER 0: PRINT AT 8,0;"type the name of the text file";AT 10,0;"and
press ENTER"
2020 LET j0=0: LET i=16: GO SUB 6000
2025 INPUT "MD (1 or 2) or c for cassette "; LINE m$: IF m$<>"1" AND m$<>"2" AND
 m$<>"c" THEN  GO TO 2025
2030 LET b=FN p(62216): IF m$<>"c" THEN  LOAD *"m";VAL m$;a$CODE (a+b),((FN p(62
221)+22)*64-a): GO TO 5
2040 LET b=FN p(62216): IF m$="c" THEN  LOAD a$CODE (a+b),((FN p(62221)+22)*64-a
): GO TO 5
```

```
3000 FOR i=23296 TO 23361: POKE i,32: NEXT i
3005 POKE 23362,NOT PI
3007 PRINT AT 3,4;"to use old word just ENTER"'AT 5,(LEN q$<27)*(26-LEN q$)/2;"o
w= '";q$;"'"
3010 PRINT AT 8,0;" or  type  a  new  word  to  be          replaced / found"
3012 LET j0=NOT PI: LET i=10: GO SUB 6000: IF a$="" THEN  LET a$=q$
3015 LET q$=a$
3020 LET j=NOT PI: FOR i=VAL "1" TO LEN a$: POKE 23297+i,CODE a$(i): IF a$(i)="
" THEN  LET j=j+1
3021 NEXT i
3022 IF j<>NOT PI THEN  CLS : PRINT AT 12,0;"just a word - no spaces allowed": G
O TO 3000
3025 POKE 23297,LEN a$
3030 PRINT AT 12,NOT PI;"with (just ENTER for find only)": LET i=14: GO SUB 6000
3040 IF a$="" THEN  POKE 23362,1: GO TO 3060
3050 FOR i=VAL "1" TO LEN a$: POKE 23329+i,CODE a$(i): NEXT i
3060 LET a=USR 64955: LET a=USR 64333: GO TO 20
4000 PRINT AT NOT PI,VAL "9";"Tasword Two";AT VAL "1",VAL "4";"? Tasman Software
 1983";AT VAL "2",VAL "6";"MICRODRIVE VERSION": RETURN
6000 LET u$=" ": LET hl=NOT PI: LET a$="": PRINT AT i,j0; FLASH 1;" "
6010 LET j=j0: IF INKEY$<>"" THEN  GO TO 6010
6020 PRINT  FLASH VAL "1";AT i,j;u$: LET b$=INKEY$
6030 IF b$="" THEN  GO TO 6020
6040 IF CODE b$=13 THEN  PRINT AT i,j;" ": RETURN
6045 IF CODE b$=14 AND hl=NOT PI THEN  GO TO 6300
6047 IF hl=VAL "1" THEN  GO TO 6400
6050 IF CODE b$<>12 THEN  GO TO 6170
6060 IF j=j0 THEN  GO TO 6200
6070 LET j=j-1: PRINT AT i,j; FLASH 1;u$; FLASH 0;" ": LET a$=a$( TO j-j0): GO T
O 6200
6170 IF CODE b$<32 OR CODE b$>127 THEN  GO TO 6200
6180 BEEP .005,5: PRINT AT i,j;b$: LET j=j+1: LET a$=a$+b$
6190 IF j=32 THEN  PRINT AT i+1,NOT PI;" ": RETURN
6200 IF INKEY$<>"" THEN  GO TO 6200
6210 GO TO 6020
6300 LET hl=VAL "1": LET u$="E": BEEP .005,5: GO TO 6200
6400 IF CODE b$=14 THEN  LET hl=NOT PI: LET u$=" ": BEEP .005,5: GO TO 6200
6405 LET h$="": GO SUB 6500+CODE b$
6410 IF h$="" THEN  GO TO 6200
6415 LET b$=h$: LET hl=NOT PI: LET u$=" ": GO TO 6180
6578 RETURN
6579 LET h$="Ö": RETURN
```

```
6583 RETURN
6584 LET h$="?": RETURN
6585 LET h$="Ü": RETURN
6588 RETURN
6589 LET h$="Ä": RETURN
6610 RETURN
6611 LET h$="ö": RETURN
6615 RETURN
6616 LET h$="ß": RETURN
6617 LET h$="ü": RETURN
6620 RETURN
6621 LET h$="ä": RETURN
6800 RETURN
7000 DEF FN p(x)=PEEK x+256*PEEK (x+1)
```

Compiler

Programme in Basic haben den Nachteil, daß sie bei umfangreichen Rechenoperationen relativ langsam ablaufen. Wesentlich schneller geht es, wenn man dem Computer die mühselige Arbeit des Übersetzens von Basic in Maschinensprache abnimmt. Das ist allerdings eine recht aufwendige und nicht ganz einfache Arbeit.

Aber auch hier kann der Computer Abhilfe schaffen, wenn ein geeignetes Programm vorhanden ist. Ein solches geeignetes Programm heißt 'Compiler' und tut genau das, was eben gewünscht wurde: es übersetzt von Basic in Maschinensprache. Ein solcher Compiler ist natürlich selbst auch in Maschinensprache geschrieben.

Der Compiler wird zunächst in den Spectrum geladen. Dann kann entweder mit der Programmierung in Basic begonnen werden oder auch ein bereits fertiges Programm in den Spectrum geladen werden und dieses mit einem einfachen Aufruf übersetzt werden.

Je nach der Art des Compilers läuft das compilierte Programm etwa 5 bis 50 mal schneller ab als in Basic. Generell kann man zwischen Compilern unterscheiden, die nur ganze Zahlen compilieren können, und solchen, die auch Fliesskommazahlen beherrschen. Vielleicht fragt man sich, was haben die Ersteren für eine Bedeutung, wenn es doch Compiler gibt, die das ganze Zahlenspektrum beherrschen.

Die Lösung liegt in der Geschwindigkeit. Gerade die Darstellung von Zahlen in Fliesskommanotation ist sehr zeitaufwendig. Aber warum soll man sich damit herumärgern, wenn dieses gar nicht nötig ist, man aber nocheinmal 10 mal schnellere Programme herstellen kann?

Möchte man also zum Beispiel Spiele, bei denen es auf Reaktionsschnelligkeit ankommen soll und bei denen die Grafik am wichtigsten ist, übersetzen, so empfiehlt sich ein

'einfacher' Compiler. In diesen Fällen können ja die gewonnenen Punkte immer als ganze Zahlen ausgegeben werden.

Soll dagegen ein Geschäftsprogramm in Maschinensprache vorliegen, wird man zum 'allgemeineren' Compiler greifen. Welcher Geschäftsmann möchte schon auf alle Pfennige verzichten.

In der Regel beherrschen die Compiler aber nicht alle Befehle, die der Spectrum zur Verfügung stellt. So ist meist die Dimensionszahl eines Feldes auf eins begrenzt. Das ist aber weiter kein Problem, da man natürlich mit etwas Aufwand jedes mehrdimensionale Feld als eindimensionales darstellen kann, indem alle Größen hintereinander gespeichert werden. Der Spectrum tut dies intern sowieso.

Weiterhin kann der Compiler in der Regel die VAL Anweisung und berechnete Sprünge nicht ausführen. Aber auch hier kann das Problem durch z.B. CODE CHR$ und ähnliche Klimmzüge umgangen werden.

Besonders angenehm ist bei der Benutzung von Compilern, daß sich beispielsweise mehrere Unterroutinen nacheinander compilieren lassen, wobei jede Routine an einer anderen Speicheradresse beginnen kann. Diese Routinen können dann in beliebigen Programmen durch RAND USR ... aufgerufen und genutzt werden.

KAPITEL 8:
G R A F I S C H E D A R S T E L L U N G E N

Einleitung

Der Spektrum verfügt über sehr gute grafische Fähigkeiten, die ja auch in vielen Spielprogrammen ansprechend ausgenutzt werden. In diesem Kapitel wollen wir uns aber nicht mit dem Malen von Bildern, sondern mit der grafischen Darstellung von Tabellen beschäftigen.

Solche Business-Grafik Programme sind beliebte Anwendungen der Personal Computer, aber auch unser Spektrum kann hier einiges bieten, und das nocht nicht einmal mit allzu großem Aufwand. Dies soll dann auch das erste Programm sein, welches eine Anwendung in einem Kleinbetrieb finden kann.

Die beiden ansprechendsten grafischen Darstellungsarten von Wertetabellen sind Balken- und Kreis- oder Kuchendiagramme. Stellen wir also ein Programm her, mit dem wir Daten auf diese beiden Arten auf den Bildschirm bringen können.

Das fertige Programm soll in der Lage sein, sich den Maßstab selbst zu wählen, die Daten von mehreren Diagrammen zu speichern und Änderungen durchführen zu lassen.

Entwicklung des Programms Busygraph

Der Aufbau des Programms ist im Prinzip sehr einfach. Wir benötigen die folgenden Routinen:

1. Initialisierung der Diagramme, d. h. Speicherplatz reservieren

2. ein Menü, welches wir gleichzeitig als Inhaltsverzeichnis für die verschiedenen Diagramme auslegen werden

3. Steuerung der Bildschirmdarstellung

4. Erzeugung der Balkendiagramme

5. Erzeugung der Kreisdiagramme

6. Durchführung von Eingaben und Änderungen

7. Abspeichern von Programm und Daten

Beginnen wir mit der Speicherplatzreservierung. Beschränken wir uns auf 15 Diagramme, die gleichzeitig im Programm abrufbar sind. Das sollte ausreichend sein. Wer mehr Diagramme speichern will, braucht das Programm nur mehrmals abzuspeichern. Also:

```
20 DIM t$(15,20)
```

d.h. jede Überschrift kann bis zu 20 Zeichen lang sein. Da die Titel noch nicht belegt sind, folgt

```
30 FOR i=1 TO 15:LET t$(i)="frei":NEXT i
```

Die maximale Anzahl von Werten, die in ein Diagramm eingegeben werden kann, soll 24 sein. Diese Größe nennen wir dimax. Weiterhin definieren wir die Größe dim, welche

später die wahre Anzahl der Werte in einem Diagramm gibt und für jedes Diagramm in der Größe a(dim) gespeichert wird. Die Werte sollen in x(dim,15) sein. Zusätzlich benötigen wir einen Hilfsvektor, in dem die Werte in einem brauchbaren Maßstab vorliegen werden.

```
40 LET dimax=24:LET dim=dimax:DIM x(dim,15)
50 DIM y(dim):DIM a(dim)
```

Diesen Teil des Programms dürfen wir nur einmal durchlaufen, da sonst alle Daten gelöscht werden. Geben wir also weiter ein

```
10 BORDER 0:PAPER 0:INK 6:LET m=9000:GO TO m
```

Dies hat den Vorteil, daß im Falle eines Crashs das Programm mit GO TO 1 gestartet werden kann, ohne Daten zu verlieren.

Nun also zum Menü, welches wir in Zeile 9000 beginnen.

```
9000 CLS:PRINT TAB 2; INK 3;"B U S Y G R A P H  Menu"'
9010 FOR i=1 TO 15: PRINT i;TAB 5;t$(i): NEXT i
9020 PRINT AT 20,0; INK 6;"E......E N D E"
9100 INPUT INK 4;"Bitte eine Zahl eingeben! ";LINE m$
```

Nun muß auf eine ordnungsgemäße Abfrage geprüft werden:

```
9110 IF m$="e" THEN GO TO 9900
9120 FOR i=1 TO 15: IF m$=STR$ i THEN GO TO 5000
9130 NEXT i
```

Wenn diese Schleife wirklich bis zum Ende durchlaufen wird, liegt eine falsche Eingabe vor und wir springen zurück nach 9100:

```
9140 GO TO 9100
```

Bei 9900 erfolgt nun die Abfrage, ob die Daten gesichert werden sollen und die Möglichkeit, auf Kassette oder

Microdrive abzuspeichern, wie dies in Kapitel 4 aufgeführt ist.

Bei 5000 wird die Darstellungsart, also Balken- oder Kreisdiagramm, gewählt und nach erfolgtem Zeichnen auf weitere Befehle gewartet. Wir können jedesmal beim Wählen der Darstellungsart diejenige wählen, die beim letzten Mal betrachtet wurde. Dazu benutzen wir der Einfachheit halber eine Größe x0, welche die Anfangskoordinate des Diagramms ist. Beim Balkendiagramm ist x0 = 40, beim Kreisdiagramm 90.

```
5000 LET dim=a(VAL m$)
5010 IF x0<50 THEN GO TO 1000
5020 IF x0>50 THEN GO TO 3000
5100 PAUSE 0: LET n$=INKEY$
5200 IF n$="m" THEN GO TO m
5300 GO TO 5100
```

Sicherheitshalber sollten wir auch noch schreiben

```
60 LET x0=40: LET y0=20
```

damit diese Größen auch beim ersten Initialisieren definiert sind.

Bei Zeile 1000 wird also das Balkendiagramm erzeugt, ab Zeile 3000 das Kreisdiagramm. In beiden Fällen wollen wir oben zunächst eine Kommandozeile eingeben, damit wir uns erinnern, welche Optionen in diesem Stadium offen sind. Da wir diese Zeile zweimal benötigen, schreiben wir sie in eine Unterroutine in Zeile 100:

```
100 PRINT INK 4;"M Balken Kreis Next Last AendernD";VALm$
110 RETURN
```

Hierbei steht M natürlich für Menü. Next und Last sollen die Möglichkeiten geben, das nächste bzw. letzte Diagramm zu sehen, ohne extra ins Menü gehen zu müssen und die entsprechende Nummer einzugeben. Das D zusammen mit VALm$

zeigt in der zweiten Zeile die Nummer des aktuellen Diagramms an.

Um diese Kommandozeile zu erhalten, schreiben wir also

```
1000 CLS: GO SUB 100
```

und weiter

```
1010 PRINT AT 2,3; INK 3;t$(VAL m$)
```

Nun muß das Diagramm gezeichnet werden. Zunächst eine Umrandung:

```
1100 LET x0=40: LET y0=20
1110 FOR i = 0 TO 1
1120 PLOT x0+i,y0+i
1130 DRAW 200,0: DRAW 0,120
1140 DRAW -200,0: DRAW 0,-120
1150 NEXT i
```

und eine Beschriftung der x-Achse:

```
1160 FOR i=INT (1+dim/6) TO dim STEP INT (1+dim/6)
1170 PRINT AT 20,(x0-8)/8+i/8*(192/dim);i: NEXT i
```

Die Werte sollen bzw. werden im Vektor x(dimax,VALm$) auf den ersten Stellen stehen. Wieviele Werte belegt sind, war ja in a(VAL m$) gespeichert und in Zeile 5010 in die Größe dim übertragen worden. Jetzt müssen wir den größten dieser Werte finden, um die Werte in einem geeigneten Maßstab darstellen zu können.

```
1180 LET max=0
1190 FOR i=1 TO dim: IF x(i,VAL m$)>max THEN LET max=
     x(i,VAL m$)
1200 NEXT i
1210 LET top=max
```

Falls max = 0 ist, wurde noch kein Wert in dieses

Diagramm eingegeben und es sollte sofort zur Routine zum Eingeben bzw. Ändern von Daten gesprungen werden. Diese soll in Zeile 4000 beginnen.

```
1220 IF max=0 THEN PRINT AT 20,0,,: GO TO 4000
```

Falls aber doch schon Daten vorhanden sind, müssen wir diese vernünftig normieren:

```
1230 LET max=100/max
1240 FOR i=1 TO dim: LET y(i)=max*x(i,VAL m$): NEXT i
```

Hier wird also der Hilfsvektor y benötigt. Um eine sinnvolle Beschriftung der y-Achse zu erreichen, machen wir einen kleinen Trick, der hinreichend gut funktioniert:

```
1250 LET sc=1: IF top<30 THEN LET sc=10
1260 FOR i=0 to 4
1270 PRINT AT 9+i*3-y0/8,0; OVER 1;INT (sc*(top-i/4*top))
     /sc
1280 NEXT i
```

Nun müssen wir lediglich noch die Balken an die richtigen Stellen bringen, und dieser Teil ist erledigt.

```
1300 LET xa=x0+50/dim
1310 FOR i=1 TO dim
1320 FOT j=0 to 125/dim STEP 2
1330 PLOT xa+j,y0
1340 DRAW 0,y(i)
1350 NEXT j
1360 PLOT xa,y(i)+y0
1370 DRAW 125/dim,0
1380 DRAW 0,-y(i)
1390 LET xa=xa+200/dim
1400 NEXT i
1900 GO TO 5100
```

Das ist soweit ja schon in Ordnung, nur testen können wir die Sache noch nicht, da das Programm ja immer zu Zeile

4000 springt, solange noch keine Daten eingegeben sind.

Ehe wir uns an die Erstellung der Kreisdiagramme begeben, erledigen wir daher kurz diesen Punkt. Zunächst benötigen wir eine neue Kommandozeile:

```
4000 PRINT AT 0,0; INK 5;"EING:";
4010 PRINT INK 3;"M Titel Werte Loeschen  OK"
4020 PAUSE 0: LET n$=INKEY$: IF n$="o" THEN GO TO 5000
4030 IF n$="t" THEN GO TO 4200
4040 IF n$="w" THEN GO TO 4300
4050 IF n$="m" THEn GO TO m
4060 IF n$="l" THEN GO TO 4100
4070 GO TO 4000
```

Da wir mit dem Löschen von Daten immer vorsichtig sein wollen, erfolgt in Zeile 4100 zunächst eine Sicherheitsabfrage:

```
4100 INPUT "Loeschen? J/N "; LINE n$
4110 IF n$<>"j" THEN GO TO 4000
4120 FOR i=1 TO dimax: LET x(i,VAL m$)=0: NEXT i
4130 LET a(VAL m$)=1
4140 LET t$(VAL m$)=" frei"
4150 GO TO 5000
```

Nun müssen wir uns um die Eingabe bzw. Änderung von Überschrift, Anzahl der Werte und der Werte selbst kümmern. Fangen wir mit der Überschrift an:

```
4200 INPUT "Eingabe: ";LINE n$: IF n$<>"" THEN LET t$(VAL
     m$)=n$
4210 PRINT AT 2,0,,;AT 2,3; INK 3;t$(VAL m$)
4220 GO TO 4000
```

Dann die Anzahl der Werte:

```
4300 INPUT "Anzahl der Werte? (ENTER=";(a(VAL m$));")  ";
     LINE n$: IF n$="" THEN GO TO 4340
4310 FOR i=1 TO dimax: IF n$=STR$ i THEN GO TO 4330
4315 NEXT i
```

```
4320 GO TO 4000
4330 LET a(VAL m$)=VAL n$
4340 LET dim=a(VAL m$)
4350 INPUT "Werte aendern? J/N ";LINE n$: IF n$<>"j" THEN
     GO TO 4000
4360 FOR i=1 TO a(VAL m$)
4370 INPUT "Wert ";(i);" ENTER=";(x(i,VAL m$));" E=ENDE "
     ; LINE n$: IF n$="" THEN GO TO 4400
4380 IF n$="e" THEN GO TO 5000
4390 LET x(i,VAL m$)=VAL n$
4400 NEXT i
```

Eigentlich sollte hier noch eine Abfrage eingebaut werden, ob der Wert korrekt eingegeben wurde. Das sei dem Leser überlassen, falls er dies wünscht.

Nun aber zum Erstellen der Kreisdiagramme. Falls noch keine Werte eingegeben wurden, darf natürlich kein Diagramm gezeichnet werden.

```
3000 LET sum=0
3010 FOR i=1 TO dim
3020 LET sum=sum+x(i,VAL m$)
3030 NEXT i
3040 IF sum=0 THEN GO TO 1000
3050 CLS: GO SUB 100
3060 PRINT AT 2,3; INK 3;t$(VAL m$)
```

Zeichnen wir nun einen Kreis:

```
3100 LET x0=90: LET y0=72
3110 CIRCLE x0,y0,y0
```

Nun normieren wir die Werte wieder im Hilfsvektor y(dim):

```
3120 FOR i=1 TO dim: LET y(i)=x(i,VAL m$)/sum: NEXT i
```

Um die Kreisausschnitte zu zeichnen, müssen wir uns an die Trigonometrie im Mathematikunterricht erinnern:

```
3200 LET win=0
3210 LET xp=22-y0/8: LET yp=x0/8
3220 LET u=2*PI*y0
```

Um die Bezeichnung der Kreissegmente richtig positionieren zu können, führen wir noch den Wert scal ein:

```
3230 LET scal=.9
```

Jetzt werden wir eine Schleife über die Anzahl der Werte machen und jeweils den Mittelpunkt des Kreises plotten und dann eine Linie mit dem geeigneten Winkel zur vorherigen Linie:

```
3400 FOT i=1 TO dim
3410 PLOT x0,y0
3420 LET win=win+y(i)
3430 LET x1=y0*COS (2*PI*win)
3440 LET x2=scal*y0*COS (2*PI*(win-y(i)/2))
3450 LET y1=y0*SIN (2*PI*win)
3460 LET y2=scal*y0*SIN (2*PI*(win-y(i)/2))
3470 DRAW x1,y1
3480 PRINT AT 21-y0/8-y2/8,x0/8-1+x2/8; INVERSE 1;i
3490 NEXT i
```

Da wir beim Kreisdiagramm auf der linken Bildschirmhälfte noch Platz haben, printen wir die prozentualen Anteile der Werte an der Summe aller Werte sowie den Mittelwert:

```
3500 PRINT AT 3,23; INK 5;"Werte (%)"
3510 FOR i=1 TO dim STEP 2: PRINT AT (i+6)/2,21; INK 5;i;
3520 PRINT TAB 23;INT (1000*y(i)))/10;
3530 IF (i+1)<=dim THEN PRINT TAB 28;INT (1000*y(i+1))/10
3540 NEXT i
3550 PRINT AT 17,20;"SUMME";AT 18,20;INT (100*sum)/100
3560 PRINT AT 20,20;"Mittelwert";AT 21,20;INT (100*(sum/d
     im))/100
3900 GO TO 5100
```

Nun wären wir fertig, wenn wir nicht noch einen kleinen

Punkt vergessen hätten. Nachdem ein Diagramm gezeichnet ist, kehrt das Programm in die Zeile 5100 zurück. Hier haben wir aber erst für die Rückkehr ins Menü gesorgt, jedoch noch nicht für die weiteren Optionen. Erledigen wir diesen Punkt noch schnell:

```
5210 IF n$="b" THEN GO TO 1000
5220 IF n$="k" THEN GO TO 3000
5230 IF n$="n" AND VAL m$<15 THEN LET m$=STR$ (VAL m$+1)
5240 IF n$="l" AND VAL m$>1 THEN LET m$=STR$ (VAL m$-1)
5250 IF n$="n" OR n$="l" THEN GO TO 5000
5260 IF n$="a" THEN GO TO 4000
5900 GO TO 5100
```

Um das Programm zum ersten Mal zu starten oder später einmal alle Werte zu löschen, geben wir GO TO 20 ein. Im folgenden geht es dann immer mit GO TO 1.

Arbeit mit dem Programm

Nun wollen wir das Programm ein wenig kennenlernen (und etwas verbessern). Geben wir, nach dem Sichern, GO TO 20 ein. Wenn alles funktioniert, sollte jetzt auf dem Bildschirm das folgende erscheinen:

```
     B U S Y G R A P H  Menu

1     frei
2     frei
3     frei
4     frei
5     frei
6     frei
7     frei
8     frei
9     frei
10    frei
11    frei
12    frei
13    frei
14    frei
15    frei

E.....ENDE

BITTE eine Zahl eingeben!
```

Tippen wir jetzt '1 ENTER' ein. Zunächst erscheint kurz die Kommandozeile

```
M Balken Kreis Next Last Aendern
```

und

```
D1
     frei
```

und die Umrandung für das Diagramm. Da aber alle Werte noch zu 0 gesetzt sind, kann kein Diagramm gezeichnet werden und das Programm weiß, daß wir nun ein neues Diagramm aufsetzen wollen. Daher zeigt es sofort die entsprechende Kommandozeile

Eing: Menu Loeschen Titel Werte Ok

an. Falls wir keine Daten eingeben wollen, geht es mit 'm' ins Menü zurück.

Drücken wir dagegen 'T', so können wir eine Überschrift eingeben, zum Beispiel 'Test'. Dies ersetzt sofort das 'frei', welches vorher dastand. Nun geben wir 'W' ein, und es erfolgt die Abfrage auf die Anzahl der Werte. Soll sie erhalten bleiben, brauchen wir nur ENTER zu tippen, sonst eine Zahl zwischen 1 und 24. Geben wir '10' ein.

Jetzt können wir angeben, ob Werte geändert werden sollen und beantworten die Frage mit 'j'. Geben wir der Reihe nach die Werte 1, 2, 3 ... bis 10 ein und schauen, was passiert. Nach der letzten Eingabe wird das Diagramm automatisch gezeichnet. Daneben wird auch die erste Kommandozeile ausgegeben.

Wir können uns diese Werte auch in Form eines Kreisdiagramms durch Drücken von 'K' ansehen. Hier werden zusätzlich noch die prozentualen Anteile der einzelnen Werte sowie die Summe und der Mittelwert gezeigt.

Wollen wir einen Wert ändern oder hinzufügen, so geschieht dies durch Drücken von 'A'. Lassen wir die Anzahl der Werte unverändert, geben aber für die erste Zahl eine 20 ein. Anschliessend tippen wir 'e' und schon ist das neue Diagramm fertig.

Was am Programm noch etwas stört, ist die Tatsache, daß wir die Werte nicht im Überblick sehen können und einen eventuell falsch eingegebenen Wert erst ändern können, wenn die Eingabe mit 'e' abgeschlossen oder der letzte

Wert erreicht wurde. Das läßt sich allerdings leicht beheben.

Unterbrechen wir mit BREAK das Programm und geben ein:

```
200 For i=5 TO 17: PRINT AT i,6;"                       "
    : NEXT i: RETURN
```

Dies erzeugt ein Feld von 13 Zeilen mit 23 Spalten.

Nun überschreiben wir die Zeile 4350 mit

```
4350 INK 0: PAPER 7: GO SUB 200: PRINT AT 5,6;"WERTETABEL
     LE"
```

und dann:

```
4352 FOR i=1 TO dim: IF i<13 THEN PRINT AT i+5,6;i;TAB 9;
     x(i,VAL m$)
4353 IF i>12 THEN PRINT AT i-7,18;i;TAB 22;x(i,VAL m$)
4354 NEXT i
4356 INK 6

4380 IF n$="e" THEN PAPER 0: INK 6: GO TO 5000
4382 IF n$="l" AND i>1 THEN LET i=i-1: GO TO 4370
4384 IF CODE n$>57 THEN GO TO 4370

4392 IF i<13 THEN PRINT AT i+5,9;PAPER 7;"        ";AT i+5,
     9;INK 0;x(i,VAL m$)
4394 IF i>12 THEN PRINT AT i-7,22;PAPER 7;"        ";AT i-7
     ,22;INK 0;x(i,VAL m$)
4396 PAPER 0
```

Nun sichern Sie bitte diese Version. Wenn Sie danach wieder ins Porgramm und eine Werteeingabe oder Änderung durchführen wollen, erscheint eine Wertetabelle, die alles recht übersichtlich gestaltet. Wenn man die Änderung aus einem Kreisdiagramm heraus macht, erscheint dies fast wie die Windowtechnik.

Bei der Eingabe kann jetzt durch Drücken von 'L' zum letzten Wert zurückgesprungen werden. Beim Betrachten der Diagramme bewirkt Drücken von 'L' bzw. 'N' den Übergang zum vorherigen bzw. nächsten Diagramm, ohne ins Menü zurück zu müssen.

Das Programm läßt viel Raum für Ergänzungen zu. So können Sie als weitere Darstellungsmöglichkeit Liniendiagramme einführen. Wenn Sie es benötigen, können Sie auch negative Zahlen zulassen. Falls Sie besonders ambitioniert sind, bauen Sie die Möglichkeit ein, zwei Diagramme gleichzeitig zu zeigen oder zwei Diagramme zu addieren und als neues Diagramm abzuspeichern.

Zum Abschluß dieses Kapitels noch einmal das gesamte Listing des Programms.

```
  10 BORDER 0: PAPER 0: INK 6: LET m=9000: GO TO m
  20 DIM t$(15,20)
  30 FOR i=1 TO 15: LET t$(i)="frei": NEXT i
  40 LET dimax=24: LET dim=dimax: DIM x(dim,15)
  50 DIM y(dim): DIM a(dim)
  60 LET x0=40: LET y0=20
  90 GO TO 1
 100 PRINT  INK 4;"M Balken Kreis Next Last AendernD";VAL m$
 110 RETURN
 200  FOR i=5 TO 17: PRINT AT i,6;"                    ": NEXT i: RETURN
1000 CLS : GO SUB 100
1010 PRINT AT 2,3; INK 3;t$(VAL m$)
1100 LET x0=40: LET y0=20
1110 FOR i=0 TO 1
1120 PLOT x0+i,y0+i
1130 DRAW 200,0: DRAW 0,120
1140 DRAW -200,0: DRAW 0,-120
1150 NEXT i
1160 FOR i=INT (1+dim/6) TO dim STEP INT (1+dim/6)
1170 PRINT AT 20,(x0-8)/8+i/8*(192/dim);i: NEXT i
1180 LET max=0
1190 FOR i=1 TO dim: IF x(i,VAL m$)>max THEN  LET max=x(i,VAL m$)
1200 NEXT i
1210 LET top=max
1220 IF max=0 THEN  PRINT AT 20,0,,: GO TO 4000
1230 LET max=100/max
1240 FOR i=1 TO dim: LET y(i)=max*x(i,VAL m$): NEXT i
1250 LET sc=1: IF top<30 THEN  LET sc=10
1260 FOR i=0 TO 4
1270 PRINT AT 9+i*3-y0/8,0; OVER 1;INT (sc*(top-i/4*top))/sc
1280 NEXT i
1300 LET xa=x0+50/dim
1310 FOR i=1 TO dim
1320 FOR j=0 TO 125/dim STEP 2
1330 PLOT xa+j,y0
1340 DRAW 0,y(i)
1350 NEXT j
1360 PLOT xa,y(i)+y0
1370 DRAW 125/dim,0
1380 DRAW 0,-y(i)
1390 LET xa=xa+200/dim
1400 NEXT i
```

```
1900 GO TO 5100
3000 CLS : GO SUB 100: REM kreisdiagramm
3005 LET sum=0
3010 FOR i=1 TO dim
3020 LET sum=sum+x(i,VAL m$)
3030 NEXT i
3040 IF sum=0 THEN  GO TO 1000
3050 CLS : GO SUB 100
3060 PRINT AT 2,3; INK 3;t$(VAL m$)
3100 LET x0=90: LET y0=72
3110 CIRCLE x0,y0,y0
3120 FOR i=1 TO dim: LET y(i)=x(i,VAL m$)/sum: NEXT i
3200 LET win=0
3210 LET xp=22-y0/8: LET yp=x0/8
3220 LET u=2*PI*y0
3230 LET scal=0.9
3400 FOR i=1 TO dim
3410 PLOT x0,y0
3420 LET win=win+y(i)
3430 LET x1=y0*COS (2*PI*win)
3440 LET x2=scal*y0*COS (2*PI*(win-y(i)/2))
3450 LET y1=y0*SIN (2*PI*win)
3460 LET y2=scal*y0*SIN (2*PI*(win-y(i)/2))
3470 DRAW x1,y1
3480 PRINT AT 21-y0/8-y2/8,x0/8-1+x2/8; INVERSE 1;i
3490 NEXT i
3500 PRINT AT 3,23; INK 5;"WERTE (%)"
3510 FOR i=1 TO dim STEP 2: PRINT AT (i+6)/2,21; INK 5;i;
3520 PRINT TAB 23;INT (1000*y(i))/10;
3530 IF (i+1)<=dim THEN  PRINT TAB 28;INT (1000*y(i+1))/10
3540 NEXT i
3550 PRINT AT 17,20;"SUMME";AT 18,20;INT (100*sum)/100
3560 PRINT AT 20,20;"Mittelwert";AT 21,20;INT (100*(sum/dim))/100
3900 GO TO 5100
4000 PRINT AT 0,0; INK 5;"Eing:";
4010 PRINT  INK 4;"M  Titel Werte Loeschen  OK"
4020 PAUSE 0: LET n$=INKEY$: IF n$="o" THEN  GO TO 5000
4030 IF n$="t" THEN  GO TO 4200
4040 IF n$="w" THEN  GO TO 4300
4050 IF n$="m" THEN  GO TO m
4060 IF n$="l" THEN  GO TO 4100
4070 GO TO 4000
```

```
4100 INPUT "Loeschen? J/N "; LINE n$
4110 IF n$<>"j" THEN  GO TO 4000
4120 FOR i=1 TO dimax: LET x(i,VAL m$)=0: NEXT i
4130 LET a(VAL m$)=1
4140 LET t$(VAL m$)="frei"
4150 GO TO 5000
4200 INPUT "EINABE: "; LINE n$: IF n$<>"" THEN  LET t$(VAL m$)=n$
4210 PRINT AT 2,3,,;AT 2,3; INK 3;t$(VAL m$)
4220 GO TO 4000
4300 INPUT "Anzahl der Werte? (ENTER=";(a(VAL m$));") "; LINE n$: IF n$="" THEN
 GO TO 4340
4310 FOR i=1 TO dimax: IF n$=STR$ i THEN  GO TO 4330
4315 NEXT i
4320 GO TO 4000
4330 LET a(VAL m$)=VAL n$
4340 LET dim=a(VAL m$)
4350 INK 0: PAPER 7: GO SUB 200: PRINT AT 5,6;"WERTETABELLE"
4352 FOR i=1 TO dim: IF i<13 THEN  PRINT AT i+5,6;i;TAB 9;x(i,VAL m$)
4353 IF i>12 THEN  PRINT AT i-7,18;i;TAB 22;x(i,VAL m$)
4354 NEXT i
4356 INK 6
4360 FOR i=1 TO a(VAL m$)
4370 INPUT "Wert ";(i);" ENTER=";(x(i,VAL m$));" E=Ende "; LINE n$: IF n$="" THE
N  GO TO 4400
4380 IF n$="e" THEN  PAPER 0: INK 6: GO TO 5000
4382 IF n$="l" AND i>1 THEN  LET i=i-1: GO TO 4370
4384 IF CODE n$>57 THEN  GO TO 4370
4390 LET x(i,VAL m$)=VAL n$
4392 IF i<13 THEN  PRINT AT i+5,9; PAPER 7;"      ";AT i+5,9; INK 0;x(i,VAL m$)
4393 IF i>12 THEN  PRINT AT i-7,22; PAPER 7;"      ";AT i-7,22; INK 0;x(i,VAL m$

4396 PAPER 0
4400 NEXT i
5000 LET dim=a(VAL m$)
5010 IF x0<=50 THEN  GO TO 1000
5020 IF x0>50 THEN  GO TO 3000
5100 PAUSE 0: LET n$=INKEY$
5200 IF n$="m" THEN  GO TO m
5210 IF n$="b" THEN  GO TO 1000
5220 IF n$="k" THEN  GO TO 3000
5230 IF n$="n" AND VAL m$<15 THEN  LET m$=STR$ (VAL m$+1)
5240 IF n$="l" AND VAL m$>1 THEN  LET m$=STR$ (VAL m$-1)
```

```
5250 IF n$="n" OR n$="l" THEN  GO TO 5000
5260 IF n$="a" THEN  GO TO 4000
5300 GO TO 5100
9000 CLS : PRINT TAB 2; INK 3;"B U S Y G R A P H  Menu"''
9010 FOR i=1 TO 15: PRINT "  ";i;TAB 5;t$(i): NEXT i
9020 PRINT AT 20,2; INK 6;"E......ENDE"
9100 INPUT  INK 4;"BITTE eine Zahl eingeben! "; LINE m$
9110 IF m$="e" THEN  GO TO 9990
9120 FOR i=1 TO 15: IF m$=STR$ i THEN  GO TO 5000
9130 NEXT i
9140 GO TO 9100
9990 CLS : PRINT AT 9,7;"Daten sichern? J/N": PAUSE 0: LET m$=INKEY$: IF m$="j"
THEN  GO TO 9996
9995 CLS : STOP
9996 CLS : ERASE "m";1;"Busygraph"
9997 SAVE *"m";1;"Busygraph" LINE 1
9998 VERIFY *"m";1;"Busygraph"
```

KAPITEL 9:

D I E L I T F A ß S Ä U L E:
W E R B U N G M I T D E M S P E C T R U M

Einleitung

Sicher haben Sie sich die Demonstrationskassette angeschaut, die mit dem Spectrum mitgeliefert wurde. Sie beinhaltet einige nette Programme, die im wesentlichen in Basic geschrieben sind und sich leicht auflisten lassen. Wir wollen uns aber jetzt nicht mit diesen Basic - Programmen beschäftigen, sondern unser Augenmerk auf die großen Buchstaben richten, die in den Mitteilungen auf dem Bildschirm erscheinen.

Im vierten Kapitel hatten wir ja schon eine kleine Basic Routine gesehen, mit der Buchstaben in doppelter Höhe geschrieben werden können. Wenn wir dieses Verfahren auch für noch größere Buchstaben verwenden wollen, wird das Programm sehr langsam.

Schneller geht es in Maschinensprache, da dem Computer die Übersetzungsarbeit aus Basic abgenommen wird. Die Routine zur Erzeugung von Großschrift auf der Demonstrationskassette ist daher auch in Machinensprache geschrieben. Sie stammt, wie alle Programme auf dem Band, von der Firma PSION. Es spricht aber nichts dagegen, daß wir uns diese Routine zu Hilfe nehmen, da sie schon einmal vorhanden ist.

Wir können sie benutzen, um zusammen mit anderen Effekten ein nettes Programm zu entwickeln, mit dem wir Mitteilungen auf dem Bildschirm erscheinen lassen können. Dies kann verwendet werden, um zum Beispiel Familienangehörigen mitzuteilen, daß sie nicht mit dem Abendessen auf Sie warten sollen. Vielleicht könnte man auch Gästen auf einer Party die neuesten Bayernwitze vorführen. Es läßt

sich sicher viel Unsinniges mit einem solchen Programm anfangen.

Natürlich kann auch ein Ladenbesitzer ein Fernsehgerät in sein Schaufenster stellen und seine Kunden über die aktuellen Sonderangebote informieren. Wenn es schön farbenfroh und bewegt zugeht, ist dies sicher ein guter Blickfang.

Das Programm sollte also die Möglichkeit bieten, Text einzugeben und zu speichern, und diesen dann in beliebiger Größe und Farbe für frei wählbare Zeit auf dem Bildschirm zu zeigen. Nennen wir das Programm AWIS als Abkürzung für Anzeigen-, Werbe- und Informationssystem.

Ein wichtiger Punkt in dem Programm ist natürlich die Großschriftroutine. Nehmen wir uns also eins der PSION Programme her, zum Beispiel 'WALL', welches das erste Programm auf der B - Seite der Kassette ist. Nachdem wir das Programm durch BREAK unterbrochen haben, sehen wir, daß nach einem CLEAR 31999 der Code 'c' an die Speicherplätze 32000 bis 32339 geladen wird.

Wir können diesen Code separat auf eine Kassette oder eine Microdrive Kassette speichern. Dies sollte mit dem Befehl SAVE "c" CODE 32000,340 geschehen. Selbst wenn Sie die 48 KByte Version des Spectrums besitzen, muß der Code an diesen Speicherplätzen sein, wenn sich ein Aufruf dieser Routine nicht zu einem Chaos entwickeln soll. In der Routine sind nämlich einige Sprünge zu absoluten Adressen. Wenn sich an diesen Adressen nicht der richtige Code befindet, passieren unschöne Dinge. In der Regel muß der gute Spectrum ausgeschaltet werden, da er nicht in den Basic Betrieb zurückkehren will. Dies läßt sich natürlich umgehen, wenn man die entsprechenden Adressen ändert. Dann kann die Routine weiter nach hinten geschoben werden und so mehr Platz für das Basic Programm geschaffen werden.

Wir wollen es hier bei der vorliegenden Version belassen

und möglichst platzsparend das restliche Programm gestalten, sodaß noch genügend Text gespeichert werden kann. Das hat auch den Vorteil, daß die 16 K Besitzer auf ihre Kosten kommen. Die Routine funktioniert folgendermaßen: der Text wird in den String p$ eingegeben. Dann wird die Größe yy gesetz, die die Zeilenzahl auf dem Schirm angibt. Nun werden noch die Werte für xs und ys festgelegt. Diese sind als Skalenfaktoren für die Schriftgröße in x- und y- Richtung zu verstehen.

Schreiben wir doch folgendes kleines Testprogramm:

```
10   CLEAR 31999
20   LOAD "c" CODE 32000,340
30   INK 0: BORDER 6: PAPER 6: CLS
40   LET p$="HALLO"
50   LET yy=10
60   LET xs=2: LET ys=4
70   GO SUB 3500
90   PAUSE 0: STOP
```

Nun müssen wir noch an die Unterroutine in Zeile 3500 kommen. Diese können wir einfach aus dem Programm 'WALL' übernehmen:

```
3500 LET xx=(256-8*xs*LEN p$)/2
3510 LET i=23306: POKE i,xx: POKE i+1,yy: POKE i+2,xs
3520 POKE i+3,ys: POKE i+4,8
```

In Zeile 3500 wird die Spaltennummer derart berechnet, daß der Text genau zentriert in der Zeile erscheint. Dann werden die Größen xx, yy, xs und ys an die Speicherplätze von 23306 an geschrieben. Von hier können sie dann von der Maschinencoderoutine gelesen und verarbeitet werden. Die Speicherplätze von 23296 an sind der Zwischenspeicher für den Drucker und können hier also gefahrlos benutzt werden.

Daher werden auch die Codes der Character des Strings p$, also der Text, den wir schreiben wollen, in diesen

Speicher hinein gePOKEd:

```
3530 LET i=i+4: LET w=LEN p$:
3540 FOR n=1 TO w: POKE i+n,CODE p$(n): NEXT n
3550 POKE i+w+1,255
3560 LET w=USR 32000
3570 RETURN
```

Zeile 3560 aktiviert die Maschinencoderoutine. Wenn alles richtig eingegeben wurde, sollte ein RUN einen Gruß in großen Buchstaben auf den Bildschirm werfen. Das PAUSE 0 ist eingefügt, damit keine Meldung nach getaner Arbeit auf dem Schirm erscheint.

Wenn dies so weit in Ordnung ist, sollten Sie ein wenig mit den Parametern, also den xs, ys usw herumexperimentieren. Geben Sie auch andere Texte ein und schauen Sie, was passiert, wenn die Schrift zu breit ist, sodaß der Text nicht auf eine Zeile paßt. Die Routine ist gut geschrieben, es kommt nicht zum Crash!

Die Programmentwicklung

So, jetzt können wir uns daran geben, aus diesen Anfängen ein ganzes Programm zu bauen. Die Unterroutine ab Zeile 3500 belassen wir, nur die ersten Zeilen sollten gelöscht werden. Beginnen wir in bekannter Weise:

```
10   LET big=3500: LET m=9000: GO TO m
```

und

```
9000 INK 0: BORDER 6: PAPER 6: CLS1
9010 LET p$="AWIS-Menu": LET yy=10: LET xs=2: LET ys=4
9020 GO SUB big
9030 PRINT AT 9,3;"1 = Aufsetzen neuer Werbung"
9040 PRINT''"   2 = Testen und/oder aendern"
9050 PRINT''"   3 = Werbung laufen lassen"
9060 PRINT''"   4 = E N D E"
9070 PRINT AT 21,2;"Bitte gewuenschte Zahl tippen"
9100 PAUSE 0: LET m$=INKEY$
9110 IF m$="1" THEN GO TO 1000
9120 IF m$="2" THEN GO TO 5000
9130 IF m$="3" THEN GO TO 2000
9140 IF m$="4" THEN GO TO 9900
9150 GO TO 9100
```

Nun müssen wir uns überlegen, welche Möglchkeiten das Programm bieten soll. Bauen wir es so auf, daß man bei der Eingabe bestimmte Bilder erzeugen kann. Jedes Bild besteht aus einer beliebigen Anzahl Zeilen, jede Zeile aus bis zu 30 Zeichen. Jede Zeile soll in beliebiger Größe dargestellt werden können. Frei wählbar sollen auch die Farbe der Schrift, des Hintergrunds und der Umrandung sein. Zwischen dem Erscheinen der einzelnen Zeilen soll eine Pause beliebiger Länge gemacht werden können. Dabei soll jede Zeile auf Wunsch blinken können und wenn eine Zeile fertig ist, soll noch als besonderer Blickfang die Umrandung in schneller Folge die Farbe wechseln können.

Um diese Möglichkeiten einfach zu verwirklichen und natürlich auch jederzeit verändern zu können, werden wir zu jeder Textzeile, die in einem Feld a$ gespeichert sein soll ein Feld q$ zuordnen. Für jede Textzeile wird es dann 10 Größen geben, die beim Ablauf des Programms die Art der Darstellung festlegen. Stellen wir uns dieses Feld q$ zusammen, um ein wenig Übersicht zu haben:

q$(1) - je nachdem ob mit dieser Zeile ein neues Bild begonnen werden soll 0 oder 1

q$(2) - speichert als Code die Länge der Textzeile

q$(3) - speichert als Code die Zeilenzahl

q$(4) - gibt die Schriftgröße an

q$(5) - je nachdem ob die Zeile blinken soll 0 oder 1

q$(6) - gibt die Länge der Pause an, ehe die nächste Zeile erscheint

q$(7) - gibt die Umrandungsfarbe, für die Dauer dieser Zeile

q$(8) - gibt die Hintergrundfarbe an

q$(9) - gibt die Schriftfarbe an

q$(10) - je nachdem ob die Umrandung einmal alle Farben zeigen soll 0 oder 1

Neben diesen Möglichkeiten wollen wir auch vorsehen, Text an verschiedenen Stellen einfügen oder löschen zu können. Auch soll während des Ablaufs eingegriffen werden und Änderungen angebracht werden können. Jede dieser Optionen wird in eine Unterroutine gesetzt. Diese Unterroutinen bezeichnen wir der Übersichtlichkeit halber mit den

Anfangsbuchstaben, die die Aufgabe andeuten.

Geben wir ein:

```
80   LET nb=100: LET te=100: LET ze=120: LET gr=130: LET
     bl=140
82   LET pa=150: LET fa=160: let va=170

90   RETURN
```

und dazu

```
5    GO SUB 80
```

Mit diesen Hilfsmitteln können wir den Ablauf der Zeilen fast schon programmieren, ohne das Aufsetzen der Zeilen durchdacht zu haben. Im folgenden wollen wir möglichst wenig Speicherplatz vergeuden, damit möglichst viele Zeilen gleichzeitig gespeichert werden können. Das bedeutet, daß häufig für die 0 ein NOT PI erscheint, oder der Wert 1 durch VAL "1" ausgedrückt wird. Aus diesem Grund soll auch nicht jedes Mal bei einer Eingabe abgefragt werden, ob diese legal ist. Wer solche Abfragen häufiger im Programm haben möchte, kann sie gut einbauen. Überhaupt kann das Programm leicht nach Belieben erweitert oder geändert werden.

Ehe es losgeht, sollten wir noch die Zeile

```
70   LET l$="                              "
```

eingeben. l$ ist also ein String mit 30 Leerstellen und ermöglicht die Abfrage, ob in einer Zeile a$(i) schon ein Text eingegeben wurde. Nun zum Programmteil, der die Wiedergabe der Zeilen steuert.

```
2000 CLS : IF a$(1)=l$ THEN GO TO m
```

Wenn kein Text eingegeben ist, geht es gleich ins Menü zurück.

```
2010 LET anf = VAL "1": LET end = d
```

d wird später auf die Anzahl der möglichen Zeilen gesetzt.

```
2020 PRINT'''"Alle Zeilen? J/N": PAUSE NOT PI: LET z$ =
     INKEY$: CLS
2030 IF z$<>"n" THEN GO TO 2100
```

Diese Abfrage ermöglicht es, nur bestimmte Zeilen, diejenigen von 'anf' bis 'end' laufen zu lassen.

```
2040 INPUT "Erste Zeile = ?"; LINE z$: IF z$="" THEN LET
     z$="1"
2050 LET anf = VAL z$: IF anf>d THEN GO TO 2040
2060 INPUT "Letzte Zeile = ?"; LINE z$: IF z$="" THEN LET
     z$=STR$ d
2070 LET end=VAL z$: IF end>d THEN GO TO 2060
2080 IF anf>end THEN GO TO 2000
```

Nun beginnt die Schleife über alle Zeilen:

```
2100 FOR z=anf TO end
```

die bei 2500 enden soll

```
2500 NEXT z
```

Um jederzeit aus dem Programmablauf, der hier ja eine Endlosschleife sein soll, herauszukommen, geben wir ein:

```
2110 IF INKEY$="m" THEN GO TO m
```

So kommt man immer durch Drücken der Taste m in das Menü.


```
2070 LET end=VAL z$: IF end>d THEN GO TO 2060
2080 IF anf>end THEN GO TO 2000
```

Nun beginnt die Schleife über alle Zeilen:

```
2100 FOR z=anf TO end
```

die bei 2500 enden soll

```
2500 NEXT z
```

Um jederzeit aus dem Programmablauf, der hier ja eine Endlosschleife sein soll, herauszukommen, geben wir ein:

```
2110 IF INKEY$="m" THEN GO TO m
```

So kommt man immer durch Drücken der Taste m in das Menü.

```
2120 IF a$(z)=l$ THEN GO TO 2100
```

Wenn die erste Leerzeile erreicht wird, beginnt das ganze Spiel aufs Neue.

Jetzt können wir endlich mit der Darstellung der einzelnen Zeilen beginnen. Zunächst werden die Farben gesetzt:

```
2130 BORDER VAL q$(7,z): PAPER VAL q$(8,z): INK VAL q$(9,
     z)
```

Dann wird die Zeile festgelegt, in der Text erscheint:

```
2200 LET yy=8*CODE q$(3,z)
```

und die Schrift

```
2210 GO SUB 135
```

Wir springen in die Unterroutine 'gr', da bei der Schrift zwei Parameter berechnet werden, xs und ys. Da dies bei der Eingabe der Zeile ebenfalls durchgeführt wird, können wir uns das erneute Eingeben dieser Programmzeilen ersparen.

Nun müssen wir den Text in das Feld p$ bringen, da unsere

Routine für Großschrift dies verlangt. Die Länge des Textes steht in q$(2,z), also

```
2220 LET p$=a$(z,1 TO CODE q$(2,z))
2230 IF q$(1,z)="1" THEN CLS
```

Bildschirm löschen, falls neues Bild verlangt wurde und

```
2240 FLASH VAL q$(5,z)
```

d.h. eventuell Blinken einstellen.

Nun gehen wir in die Unterroutine 'big', um den Text auszugeben

```
2300 GO SUB big
```

und stellen sicherheitshalber ein eventuelles Blinken ab:

```
2310 FLASH NOT PI
```

Nun fragen wir ab, ob die Umrandung alle Farben durchlaufen soll

```
2320 IF q$(10,z)="1" THEN FOR k=1 TO 7: BORDER k: PAUSE
     50: NEXT k
```

Jetzt wird die Länge der Pause festgelegt. Sie ist durch q$(6,z) gegeben. Wenn CODE q$ 6,z)=0 wäre, würde die Zeile solange stehen bleiben, bis eine Taste gedrückt wird. Das wollen wir verhindern. Also

```
2350 IF CODE q$(6,z)=0 THEN LET Q$(6,z)= CHR$ 1
2400 PAUSE 50*CODE q$(6,z)
```

Zwei Möglichkeiten sollten wir an dieser Stelle einfügen, die das Programm später beim Gebrauch wesentlich komfortabler gestalten: einmal sollten wir das Programm unterbrechen können, um vielleicht ein Bild länger stehen zu lassen, zum anderen sollten wir dann auch die Möglichkeit

für Änderungen zulassen. Das läßt sich einfach verwirklichen durch die Anweisungen

```
2410 IF INKEY$="p" THEN PRINT AT NOT PI,NOT PI;"PAUSE" NO
     T PI: PRINT AT NOT PI,NOT PI;"      ": GO TO 2410
```

Falls wir dann die Nummer der Zeile wissen wollen, können wir diese durch Drücken der Taste z sehen, wenn wir eingeben

```
2420 IF INKEY$="z" THEN PRINT AT NOT PI,NOT PI;"ZEILE ";z
     : PAUSE NOT PI: PRINT AT NOT PI,NOT PI;"           ":
     GO TO 2410
```

Nun die Möglichkeit zur Änderung einer Zeile:

```
2430 IF INKEY$="k" THEN GO TO 5200
```

Die Zeile '2500 NEXT z' hatten wir schon eingegeben. Nun geben wir ein

```
2600 GO TO 2100
```

und der Endloslauf ist gewährleistet.

Jetzt wird es aber Zeit sich um die Eingabe der Texte zu kümmern, damit endlich etwas auf dem Schirm erscheint. Als erstes sollten wir uns vergewissern, ob eventuell eingegebene Zeilen gelöscht werden sollen, damit dies nicht aus Versehen geschieht:

```
1000 CLS : PRINT'''"Alle Zeilen loeschen ? J/N": PAUSE
NOT PI: IF INKEY$="j" THEN GO TO 1100
1010 CLS : LET flag=1: GO TO 1200
```

Wenn die Flagge flag = 1 ist, sollen keine Zeilen überschrieben werden.

```
1100 CLEAR : GO SUB 20
```

Ab Zeile 20 werden für alle Zeilen wieder die Anfangswerte gesetzt. Nun starten wir eine Schleife über alle möglichen Zeilen:

```
1200 FOR z=VAL"1" TO d
1210 IF flag=1 AND a$(z)<>l$ THEN GO TO 1800
1800 NEXT z
```

Es wird also die erste leere Zeile gesucht (in Zeile 1800 steht NEXT z).

Nun werden die einzelnen Abfragen gemacht, wozu jedes Mal eine Unterroutine aufgerufen wird. Diese Routinen haben wir ja in Zeile 80 bereits zum größten Teil mit Namen versehen.

```
1220 GO SUB nb
```

Dies ist die Abfrage, ob ein neues Bild begonnen werden soll. Nun wollen wir den Text eingeben, dazu werden wir an den linken Bildschirmrand die Zeilenzahlen schreiben, sodaß man sich einfacher orientieren kann. Dies geschieht in einer Routine ab Zeile 200, die noch geschrieben werden muß, allerdings sehr einfach ist.

```
1230 GO SUB 200: GO SUB te
```

Wenn eine Leerzeile eingegeben wird, soll dies das Ende der Eingabe bedeuten:

```
1240 IF CODE q$(2,z)=0 THEN GO TO m
1250 LET a$(z,1 TO LEN t$)=t$
1260 GO SUB ze: GO SUB gr
```

Nun schreiben wir diese Zeile. Um den Fluß nicht zu unterbrechen, geschieht dies durch

```
1280 GO SUB 500
```

Ab Zeile 500 steht dann eine Schreibroutine, die auch die Routine 'big' aufruft. Die wichtigsten Parameter sind eingegeben und nun soll gefragt werden, ob noch einige Größen geändert werden. Dazu werden wir eine kleine Routine benutzen, die sinnvollerweise 'frage' genannt werden soll. Also

```
1290 GO SUB frage
```

Das Ganze wird abgeschlossen mit

```
1800 NEXT z
1900 GO TO m
```

wie vorhin schon erwähnt. Einige Kleinigkeiten fehlen zwar noch, diese werden wir aber etwas später noch einfügen.

Nun wollen wir die Initialisierung vervollständigen, also den Speicherplatz reservieren, bzw. löschen und den Parametern mehr oder weniger vernünftige Anfangswerte geben.

```
20   LET d=25
30   DIM q$(10,d): DIM a$(d,30)
40   FOR k=1 TO d: LET q$(4,k)="1": LET  q$(6,k)=CHR$  5:
     LET q$(7,k)="6": LET q$(8,k)="6": LET q$(9,k)="0"
50   LET q$(1,k)="0": LET q$(5,k)="0": LET q$(10,k)="0"
60   LET big=3500: LET m=9000: LET flag=NOT PI
84   LET frage= 300
```

Damit sind als Farbe für die Schrift schwarz, für Hintergrund und Umrandung gelb vorgesehen. Dies wie auch die anderen Größen kann natürlich geändert werden, nur muß ja etwas vorgegeben sein, für alle Fälle. So ist weiterhin gesetzt 'kein neues Bild', 'kein Blinken', Pause 5 Sekunden und keine 'Farbvariationen' der Umrandung.

Jetzt haben wir die Aufgabe, die verschiedenen Unterroutinen zu erstellen, die alle Parameter einzugeben ge-

statten. Beginnen wir mit der Routine 'frage':

```
300  PRINT AT 21,NOT PI; INK NOT PI; PAPER VAL "7";"T U H
     F Z B G P N E V Okay"
310  PAUSE NOT PI: LET n$=INKEY$
312  IF n$="u" OR n$="h" OR n$="f" THEN GO TO fa
```

alle Farben werden gemeinsam behandelt.

```
313  IF n$="z" THEN GO TO ze
314  IF n$="b" THEN GO TO bl
315  IF n$="p" THEN GO TO pa
316  IF n$="n" THEN GO TO nb
317  IF n$="e" THEN GO TO 800
318  IF n$="v" THEN GO TO va
319  IF n$="o" THEN RETURN
320  IF n$="t" THEN GO TO te
321  IF n$="g" THEN GO TO gr
330  GO TO frage
```

Je nachdem welche Taste gedrückt wird, springen wir in die entsprechende Unterroutine. Falls alles in Ordnung ist, kehren wir zum Programmteil 'Aufsetzen' zurück. Wir sollten also in diesem die Vorkehrung treffen, dann die nächste Zeile zu bearbeiten. Daher geben wir ein

```
1320 IF n$="o" THEN GO TO 1800
1330 GO TO 1280
```

Nun müssen wir uns um die Eingaben der verschiedenen Parameter kümmern. Beginnen wir mit der Routine 'nb', also der Abfrage, ob mit der Zeile ein neues Bild aufgesetzt werden soll. Diese beginnt bei Zeile 100:

```
100  LET q$(1,z)="0": PRINT AT 21,NOT PI;"NEues Bild? J/N
       (0=Menu)",: PAUSE NOT PI: LET f$= INKEY$: IF
     f$="j" THEN LET q$(1,z="1": CLS
102  IF z=VAL "1" THEN GO TO 1066
104  IF f$="n" THEN let q$(6,z-1)="1"
106  IF f$<>"j" AND f$<>"n" THEN GO TO m
```

```
108  RETURN
```

Nun zur Eingabe des Textes:

```
110  LET q$(2,z)=CHR$ 0: PRINT AT 21,MOT  PI;"Text  =",,:
     INPUT LINE t$
112  IF LEN t$>30 THEN GO TO 110
114  LET q$(2,z)=CHR$ (LEN t$)
116  RETURN
```

Jetzt kommt die Abfrage nach der Zeilennummer:

```
120  GO SUB 200: PRINT AT 21,NOT PI;"ZEILE (1-20) =",,:
     INPUT LINE n$
```

Dies druckt die Zeilennummern an die linke Bildschirmseite, dann, falls Sie diese Sicherheit wollen:

```
122 FOR s=VAL "1" TO 20: IF n$=SRT$ s THEN GO TO 128
124 NEXT s
126 GO TO 120
```

und

```
128 LET q$(3,z)=CHR$ (VAL n$): RETURN
```

Ab Zeile 130 beginnt die Routine, welche die Schriftgöße festlegt:

```
130  PRINT AT 21,NOT PI;"Schriftgroesse (1-9)",: LET f$
     =INKEY$: IF f$>"9" OR f$<"1" THEN GO TO 130
132  LET q$(4,z)=CHR$ VAL f$
135  LET ys=CODE q$(4,z): LET xs=1+INT (ys/2)
137  IF CODE q$(4,z)=2 THEN LET xs=1
139  RETURN
```

Ganz einfach wird die Möglichkeit behandelt, Zeilen blinken zu lassen:

```
140  LET q$(5,z)="0": PRINT AT 21,NOT PI;"Blinken? J/N",,
     : PAUSE NOT PI: IF INKEY$="j" THEN LET q$(5,z)="1"
142  RETURN
```

Auch die Länge der Pause ist schnell festgelegt:

```
150  PRINT AT 21,NOT PI;"Wieviel Sekunden so? ",: INPUT
     LINE n$: IF LEN n$=0 THEN GO TO 150

152  IF VAL n$<1 OR NAL n$>255 THEN GOTO 150
154  LET q$(6,z)=CHR$ VAL n$
156  RETURN
```

Die Farben für Hintergrund, Umrandung und Schrift erledigen wir in einem:

```
160  PRINT AT 21,NOT PI;"Farbe (0-7)",,:  PAUSE  NOT  PI:
     LET f$=INKEY$: IF CODE f$<48 OR CODE f$>55 THEN GO
     TO 160
162  IF n$="f" THEN LET q$(9,z)=f$
164  IF n$="u" THEN LET q$(7,z)=f$
166  IF n$="h" THEN LET q$(8,z)=f$
168  PRINT AT 21,NOT PI,,: RETURN
```

Nun noch die Abfrage auf den Wechsel der Umrandungsfarbe:

```
170  LET q$(10,z)="0": PRINT AT 21,NOT PI;"Farbvariatione
     n J/N",: PAUSE NOT PI: IF INKEY$="j" THEN LET  q$(10
     ,z)="1"
172  RETURN
```

Damit haben wir diesen Teil abgeschlossen. Es fehlen jetzt noch drei kurze Routinen, die wir bereits aufgerufen haben, nämlich diejenigen ab Zeile 200 (Schreiben der Zeilennummer an der linken Bildschirmseite), die ab Zeile 500 (Ausgeben einer Zeile) und ab Zeile 800 das Einfügen von einer Textzeile.

```
200 FOR i=11 TO 30: LET f$=STR$ i: PRINT INK NOT PI;PAPER
    VAL "7";AT i-10,NOT PI;f$(2): NEXT i: RETURN
```

```
500 LET p$=t$: GO SUB 135: LET yy= CODE q$(3,z)*8: BORDER
    VAL q$(8,z): PAPER VAL q$(7,z): INK VAL q$(9,z)
510 FLASH VAL q$(5,z): GO SUB big: FLASH NOT PI: RETURN
```

und

```
800 FOR k=d-1 TO z STEP -1
810 IF a$(k)=l$ THEN GO TO 840
820 LET a$(k+1)=a$(k)
830 FOR i=VAL "1" TO 10: LET q$(i,k+1)=q$(i,k): NEXT i
840 NEXT k
850 LET a$(z)=l$
860 LET q$(2,z)=CHR$ 0: LET q$(3,z)=CHR 0: LET q$(4,z)="1
    "
870 LET q$(5,z)="0": LET q$(6,z)=CHR$ 5: LET  q$(7,z)="6"
    LET q$(8,z)="6"
880 LET q$(9,z)="0": LET q$(10,z)="0"
890 GO TO te
```

Damit können wir Texte eingeben und diese auf dem Schirm in gewünschter Form ablaufen lassen. Jedoch müssen wir dafür sorgen, eventuelle Fehler korrigieren oder Änderungen durchführen zu können.

Dies war ja ab Zeile 5000 vorgesehen.

```
5000 CLS
5010 LET anf=VAL "1": LET end=d
```

Zunächst eine Abfrage, ob lediglich neue Zeilen hinzugefügt werden sollen:

```
5020 PRINT''"Text zufuegen? J/N": PAUSE NOT PI: IF INKEY$
     ="j" THEN LET flag=1: GO TO 1200
5030 CLS : PRINT''"Alle Zeilen? J/N": PAUSE  NOT  PI:  IF
     INKEY$<>"n" THEN GO TO 5100
5050 INPUT "Ab welcher Zeile? ";LINE z$
5060 LET anf=VAL z$
```

Wer auf Nummer sicher gehen möchte, sollte hier noch eine Sicherheitsabfrage auf die Richtigkeit der Eingabe machen. Anschliessend geht es weiter mit der Korrektur. Dabei wollen wir den Code ab Zeile 2130 nutzen:

```
5100 FOR anf TO end
5110 IF a$(z)=l$ THEN GO TO m
5120 GO SUB 2130
```

und, ganz wichtig:

```
2380 IF m$="2" THEN RETURN
```

Nun weiter mit

```
5130 PRINT AT NOT PI,NOT PI;"ZEILE ";z
5200 PRINT AT 21,NOT PI;"Aendern J/N (O = Menu,W = Werb)"
     : PAUSE NOT PI: LET f$=INKEY$: IF f$="j" THEN GO TO
     5600
5210 IF f$="o" THEN GO TO m
5220 IF f$="w" THEN LET m$="3": PRINT AT 21,NOT PI,,: GO
     TO 2130
5300 NEXT z
5310 GO TO m
```

Falls eine Änderung gemacht werden soll, müssen wir die Routine 'frage' aufrufen.

```
5600 GO SUB frage
5610 IF n$<>"t" AND n$<>"e" THEN GO TO 5120
5620 IF CODE q$(2,z)=0 THEN GOTO 5900
5630 LET a$(z)=l$
5640 LET a$(z,1 TO LEN t$)=t$
5700 GO TO 5120
5900 GO SUB 400
5910 CLS : GO TO 5120
```

Das einzige, was nun noch fehlt, ist das Sichern des Programms und damit auch der Zeilen, falls sie später einmal verwendet werden sollen. Um einen Selbststart des Pro-

gramms zu erreichen, sollte man die Zeile

```
8000 POKE 23730,255: POKE 23731,125: LOAD *"m";1;"c" CODE
```

eingeben, wenn der Spectrum mit Microdrive ausgestattet ist, bzw.

```
8000 POKE 23730,255: POKE 23731,125: LOAD "c" CODE
```

wenn mit Kassette gearbeitet wird. Die POKEs sind notwendig, um nicht mit CLEAR alle Variablen und damit auch die gespeicherten Zeilen zu löschen.

Danach kann die Speicherroutine allgemein so aussehen:

```
9900 CLS : PRINT "Sichern des Programms/ENDE"
9910 PRINT ''"1 = auf Microdrive sichern"''"2 = auf Kasse
     tte sichern"''"3 = ENDE ohne sichern"''"4 = Menu"
9920 PAUSE NOT PI: LET m$=INKEY$: IF m$="3" THEN CLS : ST
     OP
9925 IF m$="4" THEN GO TO m
9930 IF m$="2" THEN GO TO 9980
9940 IF m$<>"1" THEN GO TO 9920
9945 ERASE "m";1;"awis"
9950 SAVE *"m";1;"awis" LINE 8000
9955 REM SAVE *"m";1;"c" CODE 32256,340
9960 GO TO 9900
9980 CLS: SAVE "awis" LINE 8000
9985 CLS : PRINT "Bitte eine Taste druecken!"
9990 SAVE "c" CODE 32256,340
9995 GO TO m
```

In Zeile 9955 muß das REM entfernt werden, falls auf der entsprechenden Microdrive Kassette der Maschinencodeteil noch nicht gespeichert ist.

Damit sind wir fertig und es kann losgehen. Vielleicht werden Sie nach gewisser Zeit Lust verspüren, einige weitere Möglichkeiten in das Programm einzubauen. Zum Beispiel könnte man das Feld a$ nicht 30 Zeichen, sondern

kürzer festlegen, das automatische Zentrieren abschalten und dann in einer Zeile verschiedene Schriftgrößen haben. Sie können auch die Möglichkeit einbauen, einzelne Zeilen auf dem Bildschirm abrollen zu lassen. Dies sollte allerdings in Maschinensprache programmiert werden, damit es nicht zu langsam von statten geht. Ihrer Fantasie sind keine Grenzen gesetzt.

Aber auch in der jetzigen Form ist das Programm recht nett. Wenn Sie es laden, startet es automatisch und zeigt das Menü. Denken Sie daran, daß Sie auch während die Zeilen gezeigt werden, jederzeit den Ablauf durch Drücken der Tasten 'm' für Menü, 'p' für Pause, "z" für Zeilennummer und "k" für Korrektur unterbrechen können.

Zum Abschluß dieses Kapitels noch einmal das vollständige Listing des Programms:

```
  5 GO SUB 80
 10 LET big=3500: LET m=9000: GO TO m
 20 LET d=25
 30 DIM q$(10,d): DIM a$(d,30)
 40 FOR k=1 TO d: LET q$(4,k)="1": LET q$(6,k)=CHR$ 5: LET q$(7,k)="6": LET q$(
8,k)="6": LET q$(9,k)="0"
 50 LET q$(1,k)="0": LET q$(5,k)="0": LET q$(10,k)="0": NEXT k
 60 LET big=3500: LET m=9000: LET flag=NOT PI
 70 LET v$=" ": LET l$="                              "
 80 LET nb=100: LET te=110: LET ze=120: LET gr=130: LET bl=140
 82 LET pa=150: LET fa=160: LET va=170
 84 LET frage=300
 90 RETURN
100 LET q$(1,z)="0": PRINT AT 21,NOT PI;"Neues Bild? J/N  (0=Menu)",: PAUSE NOT
 PI: LET f$=INKEY$: IF f$="j" THEN  LET q$(1,z)="1": CLS
102 IF z=VAL "1" THEN  GO TO 106
104 IF f$="n" THEN  LET q$(6,z-1)=CHR$ 1
106 IF f$<>"j" AND f$<>"n" THEN  GO TO m
108 RETURN
110 LET q$(2,z)=CHR$ 0: PRINT AT 21,NOT PI;"Text =",,: INPUT  LINE t$
112 IF LEN t$>30 THEN  GO TO 110
114 LET q$(2,z)=CHR$ (LEN t$)
116 RETURN
120 GO SUB 200: PRINT AT 21,NOT PI;"Zeile (1-20) =",,: INPUT  LINE n$
122 FOR s=VAL "1" TO 20: IF n$=STR$ s THEN  GO TO 128
124 NEXT s
126 GO TO 120
128 LET q$(3,z)=CHR$ (VAL n$): RETURN
130 PRINT AT 21,NOT PI;"Schriftgroesse (1-9)",: LET f$=INKEY$: IF f$>"9" OR f$<
"1" THEN  GO TO 130
132 LET q$(4,z)=f$
135 LET ys=VAL q$(4,z): LET xs=1+INT (ys/2)
137 IF q$(4,z)="2" THEN  LET xs=1
139 RETURN
140 LET q$(5,z)="0": PRINT AT 21,NOT PI;"Blinken? J/N",,: PAUSE NOT PI: IF INKE
Y$="j" THEN  LET q$(5,z)="1"
142 RETURN
150 PRINT AT 21,0;"Wieviel Sekunden so?",: INPUT  LINE n$: IF LEN n$=0 THEN  GO
 TO 150
152 IF VAL n$>250 OR VAL n$<1 THEN  GO TO 150
154 LET q$(6,z)=CHR$ VAL n$
156 RETURN
```

```
160 PRINT AT 21,NOT PI;"Farbe? (0-7) ",,: PAUSE NOT PI: LET f$=INKEY$: IF CODE
f$<48 OR CODE f$>55 THEN  GO TO 160
162 IF n$="f" THEN  LET q$(9,z)=f$
164 IF n$="u" THEN  LET q$(7,z)=f$
166 IF n$="h" THEN  LET q$(8,z)=f$
168 PRINT AT 21,NOT PI,,: RETURN
170 LET q$(10,z)="0": PRINT AT 21,NOT PI;"Farbvariationen? J/N",: PAUSE NOT PI:
IF INKEY$="j" THEN  LET q$(10,z)="1"
172 RETURN
200 FOR i=11 TO 30: LET f$=STR$ i: PRINT AT i-10,NOT PI; PAPER VAL "7"; INK NOT
PI;f$(2): NEXT i: RETURN
300 PRINT AT 21,NOT PI; PAPER VAL "7"; INK NOT PI;"T U H F Z B G P N E V Okay
   "
305 LET v$=" "
310 PAUSE NOT PI: LET n$=INKEY$
311 PRINT AT 21,0,,
312 IF n$="u" OR n$="h" OR n$="f" THEN  GO TO fa
313 IF n$="z" THEN  GO TO ze
314 IF n$="b" THEN  GO TO bl
315 IF n$="p" THEN  GO TO pa
316 IF n$="n" THEN  GO TO nb
317 IF n$="e" THEN  GO TO 800
318 IF n$="v" THEN  GO TO va
319 IF n$="o" THEN  RETURN
320 IF n$="t" THEN  GO TO te
321 IF n$="g" THEN  GO TO gr
330 GO TO frage
400 FOR k=z TO d-1
402 LET a$(k)=a$(k+1)
406 FOR r=VAL "1" TO 10: LET q$(r,k)=q$(r,k+1): NEXT r
420 NEXT k
425 LET a$(d)=l$
430 RETURN
500 LET p$=t$: GO SUB 135: LET yy=CODE q$(3,z)*8: BORDER VAL q$(7,z): PAPER VAL
q$(8,z): INK VAL q$(9,z)
510 FLASH VAL q$(5,z): GO SUB big: FLASH NOT PI: RETURN
800 FOR k=d-1 TO z STEP -1
801 IF a$(k)=l$ THEN  NEXT k
810 IF a$(k)=l$ THEN  GO TO 840
820 LET a$(k+1)=a$(k)
830 FOR i=VAL "1" TO 10: LET q$(i,k+1)=q$(i,k): NEXT i
840 NEXT k
```

```
 850 LET a$(z)=l$
 860 LET q$(2,z)=CHR$ 0: LET q$(3,z)=CHR$ 0: LET q$(4,z)="1"
 870 LET q$(5,z)="0": LET q$(6,z)=CHR$ 5: LET q$(7,z)="6": LET q$(8,z)="6"
 880 LET q$(9,z)="0": LET q$(10,z)="0"
 890 GO TO te
1000 CLS : PRINT '''; FLASH 1;"Alle Zeilen loeschen? J/N": PAUSE NOT PI: IF INKE
Y$="j" THEN  GO TO 1100
1010 CLS : LET flag=1: GO TO 1200
1100 CLEAR : GO SUB 20
1200 FOR z=VAL "1" TO d
1210 IF flag=VAL "1" AND a$(z)<>l$ THEN  GO TO 1800
1220 GO SUB nb
1230 GO SUB 200: GO SUB te
1240 IF CODE q$(2,z)=0 THEN  GO TO m
1250 LET a$(z,1 TO LEN t$)=t$
1260 GO SUB ze: GO SUB gr
1280 GO SUB 500
1290 GO SUB frage
1310 IF n$="t" THEN  LET a$(z,1 TO LEN t$)=t$
1320 IF n$="o" THEN  GO TO 1800
1330 GO TO 1280
1800 NEXT z
1900 GO TO m
2000 CLS : IF a$(1)=l$ THEN  GO TO m
2010 LET anf=VAL "1": LET end=d
2020 PRINT '''"Alle Zeilen? J/N": PAUSE NOT PI: LET z$=INKEY$: CLS
2030 IF z$<>"n" THEN  GO TO 2100
2040 INPUT "Erste Zeile = ?"; LINE z$: IF z$="" THEN  LET z$="1"
2050 LET anf=VAL z$: IF anf<d THEN  GO TO 2040
2060 INPUT "Letzte Zeile = ?"; LINE z$: IF z$="" THEN  LET z$=STR$ d
2070 LET end=VAL z$: IF end>d THEN  GO TO 2060
2080 IF anf>end THEN  GO TO 2000
2100 FOR z=anf TO end
2110 IF INKEY$="m" THEN  GO TO m
2120 IF a$(z)=l$ THEN  GO TO 2100
2130 BORDER VAL q$(7,z): PAPER VAL q$(8,z): INK VAL q$(9,z)
2200 LET yy=8*CODE q$(3,z)
2210 GO SUB 135
2220 LET p$=a$(z,1 TO CODE q$(2,z))
2230 IF q$(1,z)="1" THEN  CLS
2240 FLASH VAL q$(5,z)
2300 GO SUB big
```

```
2310 FLASH NOT PI
2320 IF q$(10,z)="1" THEN  FOR k=0 TO 7: BORDER k: PAUSE 50: NEXT k
2330 IF m$="2" THEN  RETURN
2340 IF CODE q$(6,z)=NOT PI THEN  LET q$(6,z)=CHR$ 1
2390 IF CODE q$(6,z)=0 THEN  LET q$(6,z)=CHR$ 1
2400 PAUSE 50*CODE q$(6,z)
2410 IF INKEY$="p" THEN  PRINT AT 0,0;"Pause": PAUSE 0: PRINT AT 0,0;"      ": GO
 TO 2410
2420 IF INKEY$="z" THEN  PRINT AT 0,0;"Zeile ";z: PAUSE 0: PRINT AT 0,0;"
  ": GO TO 2410
2430 IF INKEY$="k" THEN  GO TO 5200
2500 NEXT z
2600 GO TO 2100
3500 LET xx=(256-8*xs*LEN p$)/2
3510 LET i=23306: POKE i,xx: POKE i+1,yy: POKE i+2,xs: POKE i+3,ys: POKE i+4,8:
LET i=i+4: LET w=LEN p$: FOR n=1 TO w: POKE i+n,CODE p$(n): NEXT n: POKE i+w+1,2
55: LET w=USR 32256: RETURN
5000 CLS
5010 LET anf=VAL "1": LET end=d
5020 PRINT '''"Text zufuegen? J/N": PAUSE 0: IF INKEY$="j" THEN  LET flag=1: GO T
O 1200
5030 CLS : PRINT '''"Alle Zeilen? J/N": PAUSE NOT PI: IF INKEY$<>"n" THEN  CLS :
GO TO 5100
5050 INPUT "Ab welcher Zeile? "; LINE z$
5060 LET anf=VAL z$
5100 FOR z=anf TO end
5110 IF a$(z)=l$ THEN  GO TO m
5120 GO SUB 2130
5130 PRINT AT NOT PI,NOT PI;"Zeile ";z
5200 PRINT AT 21,NOT PI;"Aendern? J/N (O = Menu,W = Werb)": PAUSE NOT PI: LET f$
=INKEY$: IF f$="j" THEN  GO TO 5600
5210 IF f$="o" THEN  GO TO m
5220 IF f$="w" THEN  LET m$="3": PRINT AT 21,0,,: GO TO 2130
5300 NEXT z
5310 GO TO m
5600 GO SUB frage
5610 IF n$<>"t" AND n$<>"e" THEN  GO TO 5120
5620 IF CODE q$(2,z)=0 THEN  GO TO 5900
5630 LET a$(z)=l$
5640 LET a$(z,1 TO LEN t$)=t$
5710 GO TO 5120
5900 GO SUB 400
```

```
5910 CLS : GO TO 5120
8000 POKE 23730,255: POKE 23731,125: LOAD *"m";1;"c"CODE
8015 LET flag=NOT PI
9000 INK 0: BORDER 6: PAPER 6: CLS
9010 LET p$="AWIS-Menu": LET yy=10: LET xs=2: LET ys=4
9020 GO SUB big
9030 PRINT AT 9,3;"1 = Aufsetzen neuer Werbung"
9040 PRINT ''"   2 = Testen und/oder aendern"
9050 PRINT ''"   3 = Werbung laufen lassen"
9060 PRINT ''"   4 = E N D E"
9070 PRINT AT 21,2;"Bitte gewuenschte Zahl tippen"
9100 PAUSE 0: LET m$=INKEY$
9110 IF m$="1" THEN  GO TO 1000
9120 IF m$="2" THEN  GO TO 5000
9130 IF m$="3" THEN  GO TO 2000
9140 IF m$="4" THEN  GO TO 9900
9150 GO TO 9100
9900 CLS : PRINT "Sichern des Programms/ENDE"
9910 PRINT ''"1 = auf Microdrive sichern"''"2 = auf Kassette sichern"''"3 = ENDE
 ohne sichern"''"4 = Menu"
9920 PAUSE NOT PI: LET m$=INKEY$: IF m$="3" THEN  CLS : STOP
9925 IF m$="4" THEN  GO TO m
9930 IF m$="2" THEN  GO TO 9980
9940 IF m$<>"1" THEN  GO TO 9920
9945 ERASE "m";1;"awis"
9950 SAVE *"m";1;"awis" LINE 8000
9955 REM s*"m";1;"c"CODE 32256,340
9960 GO TO 9900
9980 CLS : SAVE "awis" LINE 8000
9985 CLS : PRINT "Bitte eine Taste druecken!"
9990 SAVE "c"CODE 32256,340
9995 GO TO m
```

KAPITEL 10:
D E R S P E C T R U M I M K L E I N B E T R I E B

Allgemeines

Der ZX Spectrum im professionellen Einsatz? Das klingt vielleicht zunächst ein wenig überheblich. Betrachtet man die Situation aber realistisch, so stellt man fest, daß der Spectrum eine ganze Reihe von sinnvollen Aufgaben übernehmen kann und damit für einen 'Kleinunternehmer' viel Zeiteinsparung bedeuten kann.

Darüberhinaus bietet er natürlich wieder einmal einen preiswerten Einstieg in die Datenverarbeitung. Handelt es sich um einen Spectrum Plus oder einen normalen Spectrum mit Zusatztastatur, stehen ebenfall Microdrives und ein Drucker zur Verfügung, so wird auch ein Kleinunternehmer den Spectrum kaum mehr vermissen wollen, sofern er die geeigneten Programme besitzt.

Die häufigsten Aufgaben, die anfallen, liegen im Bereich Textverarbeitung, Kundendatei und Datenverwaltung, also Lager- und Umsatzverwaltung. Zur Textverarbeitung haben wir als Beispiel in Kapitel 4 das Programm Tasword 2 erwähnt. Hiermit lassen sich problemlos Briefe und kleine Dokumente erstellen. Auch lassen sich leicht genormte Rechnungen damit schreiben, indem zunächst einmal das Rechnungsformular aufgesetzt und dieses abgespeichert wird. Bei jeder Rechnung läßt sich dieses dann neu laden und 'ausfüllen'.

Im folgenden wollen wir ein Kundendateiprogramm und eine Lager- und Umsatzverwaltung entwickeln. Zusammen mit der Textverarbeitung Tasword 2 und dem in Kapitel 8 beschriebenen Grafikprogramm steht dann ein umfangreiches Paket zur Verfügung, welches allerlei Nutzen bringen kann. Wer seinen Spectrum wirklich professionell nutzen will, wird

sicher als Speicher die Microdrives besitzen, um nicht zu lange Wartezeiten beim Laden und Sichern der Programme zu haben. Wir wollen deshalb mit einem kleinen Steuerprogramm beginnen, welches die tägliche Arbeit sehr erleichtert, da aus einem Menü die jeweiligen Programme ausgewählt werden können. Übrigens eignet sich dies Programm natürlich auch für den privaten Gebrauch.

Das Steuerprogramm

Wenn man den Spectrum einschaltet oder NEW eingegeben hat und dann RUN und ENTER tippt, schaut der Spectrum nach, ob im Microdrivelaufwerk 1 eine Kassette eingelegt ist. Wenn dies der Fall ist, wird das Directory durchforstet und nach einem Programm 'run' gesucht. Ist dieses Programm vorhanden, wird es automatisch geladen. Wir werden also unser Steuerprogramm 'run' nennen und es mit der Option 'LINE' sichern, sodaß auch der automatische Programmstart gewährleistet ist.

Das Programm ist natürlich sehr einfach und bedarf eigentlich keiner weiteren Erklärungen.

```
100  CLEAR 65535: PAPER 0: INK 6: BORDER 0: CLS
110  PRINT INK 3; TAB 7;"H A U P T - M E N U"
120  PRINT "**********************************"
130  PRINT ''''"  1  =  Lager/Umsatzverwaltung"
140  PRINT ''"  2  =  Kundendatei"
150  PRINT ''"  3  =  Grafische Darstellungen"
160  PRINT ''"  4  =  Textverarbeitung"
170  PRINT '''"  5  =  E N D E"
200  PRINT AT 21,0,,AT 21,4; INK 4;"Bitte eine Zahl einge
     ben"
300  PAUSE 0: LET m$=INKEY$
310  IF m$="5" THEN CLS : PRINT AT 9,11;"E N D E": STOP
320  IF m$="1" THEN GO SUB 600: LOAD *"m";1;"Busyman"
330  IF m$="2" THEN GO SUB 600: LOAD *"m";1;"Busydat"
340  IF m$="3" THEN GO SUB 600: LOAD *"m";1;"Busygraph"
350  IF m$="4" THEN GO SUB 600: LOAD *"m";1;"Tasword"
400  PRINT AT 21,0; FLASH 1;"******* FALSCHE EINGABE ****
     ***"
410  FOR i=1 TO 10: BEEP .05,i: NEXT i: GO TO 200
600  CLS: PRINT AT 9,9;"Bitte warten": RETURN
```

Dieses kleine Programm sichern Sie bitte nach dem Eintippen mit dem direkten Befehl

```
SAVE *"m";1;"run" LINE 1
```

Nun müssen wir uns an die Kundendatei und die Lager - und Umsatzverwaltung heranmachen. Bei diesen beiden Programmen soll folgendermaßen verfahren werden: zunächst wird eine Beschreibung der Listings gegeben, also eine Erklärung der Variablen und Felder und die Erläuterung der einzelnen Programmabschnitte. Dann wird das komplette Listing des jeweiligen Programms gegeben, an welches sich eine 'Bedienungsanleitung' anschließt.

Wenn Sie irgendwelche Änderungen am Programm durchführen wollen, sollte es mit den Erklärungen nicht allzu schwer sein, sich in den Listings zurecht zu finden.

Die Kundendatei Busydat

Beginnen wir mit der Kundendatei. Da wir keine ganz normale Datei herstellen wollen, soll das Programm nicht nur Name und Adresse eines Kunden speichern können, sondern als kleines Extra soll die Möglichkeit bestehen, zu jedem gespeicherten Namen bestimmte Kommentare mit zu speichern. Diese Kommentare sollen in 5 Kategorien, die Sie frei wählen können, eingeteilt werden. Diese Kategorien könnten zum Beispiel den bisherigen Umsatz, die Hauptinteressen oder falls erwünscht, die Haarfaben der Kunden sein. Natürlich soll es dann auch möglich sein, nur solche Kunden aus der Datei herauszusuchen, die bestimmte Kriterien erfüllen.

Noch eine kleine Besonderheit der Datei. Die zu speichernden Kunden eines kleinen Betriebs leben in der Regel nicht quer verstreut über ganz Deutschland oder gar Europa. Vielmehr werden in der Regel nicht mehr als 5 oder 10 verschiedene Orte und Vororte zur Speicherung notwendig sein. Daher soll in dieser Datei für jeden Kunden lediglich ein Code für den Ort gespeichert werden, wobei 10 verschiedene Orte vorgesehen sind, die Sie wiederum frei wählen können. Dies spart eine Menge Speicherplatz, wird aber - wie gesagt - in den meisten Fällen ausreichen. Wer dennoch mehr Orte benötigt, kann zum Beispiel zwei verschiedene Versionen des Programms speichern.

Die Struktur des Programms ist derart gehalten, daß alle Namen und Daten mit dem Programm zusammen in den Spectrum geladen werden. Dies ermöglicht relativ kleine Zugriffszeiten, aber es hat auch den Vorteil, daß jemand, der keine Microdrives besitzt, genauso mit dem Programm arbeiten kann. Er hat lediglich den Nachteil, das Steuerprogramm nicht verwenden zu können und die längeren Ladezeiten in Kauf nehmen zu müssen.

Nun zum Programm selbst. Es beinhaltet ein Menü, welches in Zeile 9000 beginnt. Hier werden die Möglichkeiten

gegeben, Daten einzugeben, zu lesen, zu korrigieren und zu löschen. Als weitere Punkte kommen die Eingabe der Stammdaten, also der Kategorien für Kommentare, die Kommentare selbst und die Orte mit Postleitzahlen, die aufgenommen werden sollen.

Die erste Zeile des Programms setzt die Farben für Vorder- und Hintergrund. Diese können Sie natürlich jederzeit nach Ihren speziellen Wünschen verändern, allerdings sollten Sie dann auch die entsprechenden Anweisungen im weiteren Programm mit ändern.

Ab Zeile 20 wird der Speicherplatz für die einzelnen benötigten Felder reserviert. Die Zeilen 200 bis 999 beinhalten verschiedene Unterroutinen, die in den einzelnen Abschnitten benötigt werden. Ab Zeile 1000 wird die Eingabe eines Kunden gehandhabt. Dabei wird zwischen Herr, Frau und Firma unterschieden. Von jedem Kunden kann Vorname, Nachname, der Code für den Wohnort sowie eine Kundennummer gespeichert werden. Anschließend kann er dann gemäß den 5 Kategorien eingeordnet werden, wobei 4 als String vorgesehen sind und die fünfte eine Zahl ist. Dies ist der Platz, an dem zum Beispiel ein Konto für den Kunden stehen kann.

Das Lesen der Kundendatei, welches ab Zeile 2000 behandelt wird, beginnt mit einem Untermenü. Hier können Sie sich entscheiden, ob Sie alle Namen lesen oder eine Auswahl treffen wollen. Sie können einen bestimmten Namen suchen, eine Kundennummer, Kunden mit bestimmten Kommentaren oder ab einer Laufnummer. Auch beim Lesen haben Sie jederzeit die Möglichkeit der Korrektur eines Eintrags. Wenn Sie dies wünschen, springt das Programm in die Zeilen ab 3000, in denen die eigentlichen Korrekturen durchgeführt werden.

Ab Zeile 4000 wird das Löschen eines Eintrags gemacht. Dies ist einfach durchgeführt, alle folgenden Einträge werden einfach einen Platz heruntergesetzt.

k$(a,4)   Code für die Kommentare der ersten vier Kategorien

Nun das Listing des Programms:

```
 5 BORDER 0: PAPER 0: INK 6
 10 LET m=9000: GO TO m
 20 LET a=100
 30 DIM q$(a): DIM n$(a,18)
 40 DIM v$(a,15)
 50 DIM s$(a,22)
 60 DIM p$(10,20)
 65 FOR i=1 TO 10: LET p$(i,5 TO 8)="frei": NEXT i
 70 DIM k(a)
 80 DIM t$(5,10): DIM k$(a,4): DIM z(a)
 85 FOR l=1 TO 5: LET t$(l)=" frei": NEXT l
 90 DIM c$(4,4,10)
100 GO TO m
200 LET f$="0": CLS : PRINT TAB 5;"A U S W A H L"''
205 PRINT "1 = Namen suchen"''"2 = Kundennr suchen"''"3 = nach Kommentar suchen
"''"4 = ab einer Laufnummer"''"5 = Menu"
210 PAUSE 0
215 IF INKEY$="5" THEN  GO TO m
220 IF INKEY$="1" THEN  GO TO 2130
225 IF INKEY$="2" THEN  GO TO 330
230 IF INKEY$="4" THEN  GO TO 300
235 IF INKEY$="3" THEN  GO TO 8000
240 GO TO 210
260 CLS : LET g$="Frau "
264 IF n$(i,1)="h" THEN  LET g$="Herr "
265 IF n$(i,1)="b" THEN  LET g$="Firma "
269 PRINT AT 0,5; INVERSE 1;i;AT 4,0; INVERSE 0;g$
270 LET k=i: GO SUB 500: PRINT AT 5,0;v$(i,1 TO l);" ";
275 LET k=i: GO SUB 510: PRINT  INVERSE 1;n$(i,2 TO l)
280 PRINT AT 7,0;s$(i)
290 PRINT p$(VAL q$(i))
292 PRINT AT 1,0;"Kundennummer",k(i);AT 10,0;"Kommentar:"
293 GO SUB 560
295 RETURN
300 INPUT "Ab Nr? (1 - ";(a);")",m$
305 FOR l=1 TO LEN m$: IF CODE m$(l)<48 OR CODE m$(l)>57 THEN  GO TO 300
310 NEXT l
315 IF VAL m$>a THEN  GO TO 300
320 LET anf=VAL m$
325 GO TO 2220
330 INPUT "Welche Nr? ";m$
335 FOR i=1 TO a: IF STR$ k(i)=m$ THEN  GO TO 350
```

```
 340 NEXT i
 345 CLS : PRINT AT 10,8; FLASH 1;"nicht vorhanden": PAUSE 150: GO TO 9000
 350 GO SUB 260
 360 GO TO 700
 500 FOR l=LEN v$(k) TO 1 STEP -1: IF v$(k,l)<>" " THEN  GO TO 505
 504 NEXT l
 505 RETURN
 510 FOR l=LEN n$(k) TO 1 STEP -1: IF n$(k,l)<>" " THEN  GO TO 515
 514 NEXT l
 515 RETURN
 550 FOR l=1 TO 12: PRINT AT 9+l,0,,: NEXT l: RETURN
 560 FOR l=1 TO 4: PRINT AT 10+l,0;l;TAB 3;t$(l);: IF k$(i,l)>" " THEN  PRINT TA
B 20;c$(l,VAL k$(i,l))
 562 NEXT l
 565 PRINT AT 15,0;"5";TAB 3;t$(5);TAB 20;z(i)
 570 RETURN
 600 FOR l=1 TO 10: PRINT AT l+10,0;l;TAB 3;p$(l): NEXT l
 602 INPUT "Code fuer Ort (N=neuer Ort)";m$
 604 LET q$(k)=m$: GO SUB 550
 606 PRINT AT 8,0;p$(VAL q$(k))
 610 RETURN
 620 REM korrektur kommentar
 630 INPUT "Nr der Kategorie?(1-5)";m$: IF CODE m$(1)>53 OR CODE m$(1)<49 THEN
GO TO 632
 635 FOR j=16 TO 21: PRINT AT j,0,,: NEXT j
 636 IF m$="5" THEN  INPUT "Zahl = ";z(k): GO TO 3300
 640 FOR j=1 TO 4: PRINT AT 17+j,0;j;TAB 3;c$(VAL m$,j): NEXT j
 650 INPUT "Code Kommentar (0=Keiner)"; LINE x$: IF x$="0" THEN  LET k$(k,VAL m$
)=" ": GO TO 3300
 660 IF x$<"1" OR x$>"4" THEN  GO TO 650
 670 LET k$(k,VAL m$)=x$
 690 GO TO 3300
 700 PRINT AT 21,0; INK 4;"Menu.....Korrektur......Loeschen"
 710 PAUSE 0
 715 IF INKEY$="l" THEN  GO TO 4100
 720 IF INKEY$="k" THEN  GO TO 3100
 730 IF INKEY$="m" THEN  GO TO m
 740 PRINT AT 21,0,,
 745 PRINT AT 21,0;"
 750 GO TO 700
 800 REM neuer ort
 810 FOR i=1 TO 10
```

```
820 IF p$(i,1)=" " THEN  GO TO 840
825 NEXT i
830 CLS : PRINT  FLASH 1;"Keine weiterer Ort zulaessig": PAUSE 150
839 GO TO 600
840 LET c$(k)=STR$ i
845 LET p$(i)=m$
850 PRINT AT 8,0;p$(i)
855 FOR i=1 TO 11: PRINT AT 10+i,0,,: NEXT i
860 GO TO 610
1000 CLS : REM *********EINGABE
1020 FOR k=1 TO a
1030 IF n$(k,1)=" " THEN  GO TO 1100
1040 NEXT k
1050 PRINT AT 10,8; FLASH 1;"Datei voll": PAUSE 300
1070 GO TO 9000
1100 INPUT "Herr/Frau/Firma  H/F/B";m$
1101 IF m$="h" THEN  LET n$(k,1)="h": GO TO 1106
1102 IF m$="f" THEN  LET n$(k,1)="f": GO TO 1106
1103 IF m$="b" THEN  LET n$(k,1)="b": GO TO 1106
1104 GO TO 1100
1106 LET g$="Frau ": IF n$(k,1)="h" THEN  LET g$="Herr "
1107 IF n$(k,1)="b" THEN  LET g$="Firma "
1108 PRINT AT 5,0;g$;
1109 INPUT "Vorname = ";v$(k): GO SUB 500
1110 PRINT v$(k,1 TO 1);" ";
1112 INPUT "Nachname = ";n$(k,2 TO )
1113 IF CODE n$(k,2)>95 THEN  LET n$(k,2)=CHR$ (CODE n$(k,2)-32)
1115 PRINT n$(k,2 TO )
1140 INPUT "Str +   = ";s$(k): PRINT AT 7,0;s$(k)
1150 GO SUB 600
1160 INPUT "Kundennr = ";k(k)
1165 PRINT AT 0,0;"Kundennummer = ";k(k)
1170 PRINT AT 12,0;"Kommentar:": FOR i=1 TO 4: PRINT i;TAB 3;t$(i): NEXT i
1175 FOR i=1 TO 4: IF k$(k,i)>"0" THEN  PRINT AT 12+i,20;c$(i,VAL k$(k,i))
1176 NEXT i
1179 PRINT AT 17,0;"5  ";t$(5);TAB 20;z(k)
1180 INPUT "Nr der Kategorie (0=Ende)"; LINE m$: IF m$="0" THEN  GO TO 1300
1190 IF m$<"1" OR m$>"5" THEN  GO TO 1180
1200 IF m$="5" THEN  INPUT "Zahl ";z(k): PRINT AT 17,20;z(k): GO TO 1180
1210 FOR i=1 TO 4: PRINT AT 17+i,3; PAPER 7; INK 0;i;" ";c$(VAL m$,i): NEXT i
1220 INPUT "Code = (0=keiner) ";x$: IF x$="0" THEN  GO TO 1170
1230 IF x$<"1" OR x$>"4" THEN  GO TO 1220
```

```
1240 LET k$(k,VAL m$)=x$
1245 FOR i=18 TO 21: PRINT AT i,0;"                              ": NEXT i
1250 GO TO 1170
1300 INPUT "Noch ein Eintrag? J/N "; LINE m$
1302 IF m$<>"j" THEN  GO TO 9000
1305 CLS
1310 LET k=k+1
1320 IF k>a THEN  GO TO 1050
1330 IF n$(k,1)<>" " THEN  GO TO 1310
1400 CLS
1410 GO TO 1100
2000 CLS : REM **********LESEN
2010 LET k=1
2100 PRINT AT 10,8;"Alle Namen? J/N": PAUSE 0
2120 IF INKEY$<>"n" THEN  GO TO 2200
2125 GO TO 200
2130 CLS : INPUT "gesuchter Name = ";r$
2132 LET anf=1
2135 FOR i=anf TO a
2140 IF r$=n$(i,2 TO 1+LEN r$) THEN  GO TO 2160
2145 NEXT i
2150 CLS : PRINT AT 10,5; FLASH 1;" Name nicht vorhanden ": PAUSE 200: GO TO m
2160 GO SUB 260
2165 LET anf=i+1
2170 PRINT AT 21,0;"Weitersuchen? J/N": PAUSE 0: PRINT AT 21,0,,
2172 IF INKEY$="j" THEN  GO TO 2135
2175 IF m$="4" THEN  GO TO 4100
2176 IF m$="3" THEN  GO TO 3100
2180 PRINT AT 21,0; INK 4;"Menu.....Korrektur......Loeschen"
2183 PAUSE 0
2185 IF INKEY$="l" THEN  GO TO 4100
2186 IF INKEY$="k" THEN  GO TO 3100
2188 IF INKEY$="m" THEN  GO TO m
2190 PRINT AT 21,0,,
2195 GO TO 2400
2210 LET anf=1
2220 FOR i=anf TO a
2250 CLS : IF n$(i,1)=" " THEN  GO TO m
2260 GO SUB 260: PAUSE 99
2360 IF INKEY$<>"" THEN  PRINT AT 20,0;"ENTER = weiter     oder....": GO TO 2180
2400 NEXT i
2999 GO TO 9000
```

```
3000 CLS : REM *******KORREKTUR
3010 GO TO 2130
3100 PRINT AT 16,0; INK 2;"Was soll geaendert werden?       ": PRINT  INK 5;"N =
Nachname","V = Vorname","S = Strasse,Nr","O = Ort+Plz","G = Herr/Frau",
3110 PRINT  INK 5;"K = Kundennr","C = Kommentar","M = Menu"
3120 PRINT  INK 2;"R = alles in Ordnung             "
3130 PAPER 0: PAUSE 0
3140 IF INKEY$="m" THEN  GO TO m
3150 IF INKEY$="n" THEN  INPUT "Neuer Nachname = ";n$(i,2 TO )
3160 IF INKEY$="v" THEN  INPUT "Neuer Vorname = ";v$(i)
3170 IF INKEY$="v" THEN  INPUT "Neuer Vorname = ";v$(i)
3180 IF INKEY$="s" THEN  INPUT "Neue Strasse+Nr = ";s$(i)
3190 IF INKEY$<>"g" THEN  GO TO 3230
3200 INPUT "Herr/Frau/Firma(H/F/B) ?"; LINE x$
3210 IF x$<>"h" AND x$<>"f" AND x$<>"b" THEN  GO TO 3200
3220 LET n$(i,1)=x$
3230 IF INKEY$="r" THEN  LET anf=i: GO TO 2220
3240 IF INKEY$="k" THEN  INPUT "Neue Kundennummer = ";k(i)
3250 IF INKEY$="c" THEN  GO TO 620
3260 IF INKEY$="o" THEN  GO SUB 550: GO SUB 600
3300 GO SUB 260: GO TO 3100
4000 CLS : REM *****LOESCHEN
4010 GO TO 2130
4100 PRINT AT 20,0,,AT 21,0;"Loeschen? J/N",,: PAUSE 0
4110 IF INKEY$="j" THEN  GO TO 4200
4120 GO TO 2260
4200 FOR k=i+1 TO a
4210 LET n$(k-1)=n$(k)
4220 LET s$(k-1)=s$(k)
4230 LET k(k-1)=k(k)
4240 LET k$(k-1)=k$(k)
4250 LET v$(k-1)=v$(k)
4260 LET q$(k-1)=q$(k)
4270 IF n$(k,1)=" " THEN  GO TO 4300
4280 NEXT k
4300 LET n$(a,1)=" "
4310 LET anf=i: GO TO 2260
5000 CLS : PRINT "Aufsetzen der Grunddaten"
5002 PRINT ''"1 = Orte mit Pltz"
5004 PRINT ''"2 = Kategorien fuer Kommentare"
5006 PRINT ''"3 = Kommentare"
5008 PRINT ''"4 = Menu"
```

```
5010 PAUSE 0: LET m$=INKEY$: IF m$="4" THEN  GO TO m
5020 IF m$<>"1" THEN  GO TO 5040
5021 CLS : PRINT "Orte und Pltz"
5022 FOR i=1 TO 10: PRINT AT 2*i,0;i;AT 2*i,3;p$(i): NEXT i
5025 INPUT "Pltz + Ort (0=ENDE)= "; LINE x$: IF x$="0" THEN  GO TO 5000
5026 PRINT AT 21,0; INVERSE 1;x$
5027 INPUT "Nummer hierfuer? (0=keine)"; LINE m$: IF m$="10" THEN  GO TO 5030
5028 IF m$<"0" OR m$>"9" THEN  GO TO 5027
5030 LET p$(VAL m$)=x$
5039 GO TO 5021
5040 IF m$<>"2" THEN  GO TO 5060
5041 CLS : PRINT "Kategorien der Kommentare"
5042 FOR i=1 TO 5: PRINT AT 2*i,0;i;AT 2*i,3;t$(i): NEXT i
5045 INPUT "Name Kategorie (0=Ende) "' LINE x$: IF x$="0" THEN  GO TO 5000
5046 PRINT AT 21,0,,AT 21,0; INVERSE 1;x$
5047 INPUT "Nummer hierfuer (0=keine) ";m$: IF m$="0" THEN  GO TO 5041
5048 IF m$<"1" OR m$>"5" THEN  GO TO 5047
5050 LET t$(VAL m$)=x$
5059 GO TO 5041
5060 IF m$<>"3" THEN  GO TO 5000
5061 CLS : PRINT "Kommentare eingeben": FOR i=1 TO 5: PRINT AT i,0;i;AT i,3;t$(i
): NEXT i
5065 INPUT "Welche Kategorie? (0=Ende)"; LINE m$: IF m$="0" THEN  GO TO 5000
5066 IF m$<"1" OR m$>"5" THEN  GO TO 5065
5067 PRINT AT VAL m$,3; FLASH 1;t$(VAL m$)
5068 IF m$="5" THEN  PRINT ''"Hier wird jeweils eine Zahl beimEingeben eines Nam
en eingegeben.Z. B. der Umsatz etc.": PAUSE 500: GO TO 5061
5070 FOR i=1 TO 4: PRINT AT 10+i,0;i;AT 10+i,3;c$(VAL m$,i): NEXT i
5072 INPUT "Kommentar = (0=Ende) "; LINE x$: IF x$="0" THEN  GO TO 5061
5074 PRINT AT 21,0;x$
5076 INPUT "Welche Nummer (0=keine) ";o$: IF o$="0" THEN  PRINT AT 21,0,,: GO TO
 5072
5078 IF o$<"1" OR o$>"4" THEN  GO TO 5076
5080 LET c$(VAL m$,VAL o$)=x$
5090 PRINT AT 21,0,,: GO TO 5070
6000 CLS : REM **************END
6010 PRINT AT 10,0;"Daten sichern? J/N": PAUSE 0
6020 IF INKEY$="n" THEN  GO TO 6200
6030 CLS : PRINT "  Daten sichern"''''"1  =  Microdrive"''''"2  =  Cassette"''''"3
 =  M E N U"''''"4  =  E N D E": LET m$=INKEY$
6040 IF m$="1" THEN  GO TO 9996
6050 IF m$="2" THEN  GO TO 6100
```

```
6055 IF m$="3" THEN  GO TO m
6060 IF m$="4" THEN  GO TO 6200
6070 GO TO 6030
6100 CLS : INPUT "Name fuer Datei = ";x$: IF x$="" THEN  GO TO 6100
6110 SAVE x$ LINE 1
6120 PRINT AT 15,0;"Tape zurueckspulen zum VERIFY","dann PLAY druecken!": VERIFY
 x$: GO TO 6030
6200 LOAD *"m";1;"run"
8000 CLS : PRINT "Kategorien": REM suchen
8010 INK 0: PAPER 5: FOR i=1 TO 5: PRINT AT 5*i-4,0;i;TAB 3;t$(i): NEXT i: PAPER
 4
8020 FOR i=1 TO 4
8030 FOR l=1 TO 4
8040 PRINT AT 5*i-5+l,18;l;" ";c$(i,l)
8050 NEXT l
8060 NEXT i
8065 PAPER 0: INK 6
8070 INPUT "Kategorie (0=ENDE)";x$: IF x$="0" THEN  GO TO m
8080 IF x$<"1" OR x$>"5" THEN  GO TO 8070
8100 INPUT "Kommentar = (0=Keins)"; LINE o$: IF o$="0" THEN  GO TO 8070
8110 IF o$<"1" OR o$>"4" THEN  GO TO m
8120 PRINT AT 5*VAL x$-5+VAL o$,20; FLASH 1;c$(VAL x$,VAL o$)
8130 PAUSE 100
8300 FOR i=1 TO a
8310 IF n$(i,1)=" " THEN  GO TO m
8320 IF k$(i,VAL x$)=o$ THEN  GO SUB 260: PAUSE 99: IF INKEY$<>"" THEN  PAUSE 0
8330 NEXT i
9000 CLS : PRINT  INK 3;TAB 3;"A D R E S S E N K A R T E I","******************
*************",,TAB 12; INVERSE 1;" ? pcb "
9010 LET y=5: PRINT AT y,y;"1 = Eingabe von Daten"
9020 PRINT AT 7,y;"2 = Lesen von Daten"
9030 PRINT AT 9,y;"3 = Korrektur von Daten"
9040 PRINT AT 11,y;"4 = Loeschen von Daten"
9050 PRINT AT 13,y;"5 = Eingabe von Stammdaten"
9060 PRINT AT 15,y;"6 = E N D E"
9070 PRINT AT 21,4;"Bitte eine Zahl eingeben"
9100 PAUSE 0: LET m$=INKEY$
9110 IF CODE m$<49 OR CODE m$>54 THEN  GO TO 9100
9120 GO TO 1000*VAL m$
9996 ERASE "m";1;"Busydat"
9997 SAVE *"m";1;"Busydat" LINE 1
9998 VERIFY *"m";1;"Busydat": GO TO m
```

Wenn Sie dies Programm eingetippt und gesichert haben, wird es das erste Mal mit GO TO 20 gestartet. Danach wird es immer mit GO TO 1 begonnen, also beim Sichern sollte es mit LINE 1 gespeichert werden. Wenn Sie RUN eingeben, sind natürlich alle gespeicherten Daten verloren.

Sie finden nach GO TO 20 das Menü auf dem Bildschirm. Geben Sie am besten zunächst die Stammdaten ein, also die Orte mit Postleitzahlen, die Namen der Kategorien und die Kommentare. Wenn Sie noch nicht alle Namen oder Orte festgelegt haben, macht das überhaupt nichts, da Sie dies später ergänzen oder ändern können.

Für die Kategorie 5 können Sie natürlich keinen Kommentar eingeben, da hier eine Zahl verlangt wird. Das Programm sagt Ihnen das aber, falls Sie es doch versuchen. Wenn Sie diese Punkte erledigt haben, können Sie wieder in das Haupt-Menü zurückkehren und beginnen, Namen einzugeben.

Dies sollte alles keine Schwierigkeiten bereiten, da bei jeder Eingabe die Möglichkeiten aufgezeigt werden. Wenn Sie Daten lesen wollen, so können Sie entweder alle Daten verlangen oder eine Auswahl treffen. Auch in diesen Fällen führt das Programm den Benutzer so, daß es keine Probleme geben sollte. Wenn die Namen gelesen werden, bleibt jeder Eintrag für einige Zeit auf dem Schirm, ehe automatisch zum nächsten übergegangen wird. Möchten Sie einen bestimmten Eintrag länger sehen oder korrigieren, drücken Sie einfach die ENTER Taste, und der Ablauf wird unterbrochen. Zusätzlich erscheint unten am Bildschirm eine Kommandozeile, die Ihnen die nun erlaubten Optionen anbietet. Sie können jetzt zum Beispiel auch Korrekturen anbringen oder Daten löschen, oder natürlich das Lesen fortsetzen bzw. zum Hauptmenü zurückkehren.

Wenn Sie die Arbeit am Programm beendet haben, werden Sie

gefragt, ob die Daten gesichert werden sollen. Zu diesem Zeitpunkt können Sie natürlich auch Sicherheitskopien anfertigen. Nachdem dies erledigt und kein Sichern mehr erforderlich ist, wird automatisch das Menüprogramm, welches wir am Beginn des Kapitels besprochen haben, geladen. Dann können Sie zum Beispiel das Lager- und Umsatzverwaltungsprogramm bearbeiten. Dazu müßte dies Programm aber erst einmal vorhanden sein. Dies wollen wir nun in Angriff nehmen.

Das Programm - es wird Busyman heissen - soll folgende Aufgaben erledigen: der Warenbestand, der aus bis zu 750 Artikelarten bestehen kann, soll, klassifiziert nach Warengruppe und Nummer in dieser Warengruppe, mit den entsprechenden Einkaufspreisen gespeichert und verwaltet werden. Zunächst kann also jederzeit abgerufen werden, wieviel Artikel einer Sorte vorhanden sind, natürlich auch welchen Wert dieser Bestand darstellt. Weiterhin soll eine Übersicht über die einzelnen Warengruppen und den Gesamtbestand gegeben werden.

Um jederzeit auf dem laufenden zu sein, wird am Ende eines Arbeitstages die Liste der Verkäufe eingegeben. Von jedem verkauften Artikel wird Warengruppe, Artikelnummer und Verkaufspreis gespeichert. Diese Daten werden dann benutzt, die verkauften Artikel aus dem Warenbestand herauszuziehen. Damit kann jederzeit der aktuelle Warenbestand erfragt werden.

Über den Verlauf eines Monats soll von jedem Artikel Buch über die Zahl der Verkäufe geführt werden. Damit läßt sich auch überprüfen, welche Artikel 'gut gehen' und welche Ladenhüter sind. Zum Ende des Monats wird der Monatsabschluß durchgeführt, wobei der erzielte Umsatz und der Gewinn gespeichert werden. Dies geschieht über den Verlauf des gesamten Geschäftsjahres. Diese Zahlen können ebenfalls jederzeit im Programm abgerufen werden.

Natürlich empfiehlt es sich, Umsatz und Gewinn und andere Daten in das Grafikprogramm einzugeben, sodaß auch eine bildliche Darstellung dieser Daten gewährleistet ist.
Nachdem in der ersten Zeile wieder die Farben festgelegt sind, wird m zu 9000 gesetzt, die Startadresse für das Menü. Dieses bietet die Auswahl zwischen 'Tagesgeschäft', Bestands- und Umsatzübersicht, Eingabe und Änderung von Daten zum Bestand, der Datumseingabe und natürlich dem

Ende der Arbeit.

Außer bei der Datumsabfrage, die sich von selbst erklärt, gelangt man nach dem Wählen eines Punktes in ein Untermenü, welches die zur Verfügung stehenden Möglichkeiten anzeigt. So gibt es beim Punkt 'Tagesgeschäft' die Punkte: Verkäufe eingeben, Liste der Verkäufe sehen und die Tagesabrechnung durchführen. Bei diesem letzten Punkt werden die Tagesverkäufe aus dem Lagerbestand herausgenommen, sodaß die Bestandsliste auf dem neuesten Stand ist. Dieser Punkt wird von Zeile 1000 an bewältigt. Alle notwendigen Unterroutinen sind wieder in den Zeilen von 20 bis 999 untergebracht.

Ab Zeile 2000 werden der Bestand und der Umsatz verwaltet. Die einzelnen Möglichkeiten werden in einem Untermenü gezeigt und sind Bestandsübersicht, Monatsübersicht, Jahresübersicht sowie die Durchführung eines Monatsabschlusses.

Von Zeile 5000 an wird die Eingabe der Stammdaten sowie die Änderung von Beständen zum Beispiel durch Retoursendungen, Umtäusche oder Nachlieferungen erledigt. Bei Zeile 6000 schließlich wird das Sichern der Daten und die Rückkehr zum Programm 'run' erledigt.

Die wesentlichen Variablen und Felder im Programm sind die folgenden:

max	gibt die maximale Anzahl an Warentypen, die gespeichert werden können
b$(max,2)	gibt für jeden Artikeltyp die Warengruppe und die Artikelnummer
w$(wg,12)	gibt die Namen der Warengruppen
wg	Anzahl der Warengruppen
e(max)	gibt den Einkaufspreis eines Artikels

z(max)	beinhaltet den Lagerbestand der Artikel
y$(max)	speichert die im laufenden Monat verkauften Mengen jedes Artikels
anz	bestimmt die maximale Anzahl an Waren, die pro Tag als Verkäufe eingegeben werden können
g$(anz,2)	speichert die Warengruppen und Artikelnummern der Tagesverkäufe
v(anz)	gibt die zugehörigen Verkaufspreise an
o(wg)	ist eine Hilfsgröße, die im Speicher bestimmt, an welcher Stelle die einzelnen Warengruppen gespeichert sind. Dies hilft, schneller auf die einzelnen Artikel zugreifen zu können
mwst	speichert den Mehrwertsteuersatz
j(12,6)	gibt die im laufenden Monat und im laufenden Jahr erzielten Verkaufszahlen, Umsätze und Gewinne
r(wg,3)	ist ein Zwischenspeicher für Verkaufszahlen, Ein- und Verkaufspreise für die Tagesabrechnung
u(wg,3)	wie r(wg,3), jedoch für den Monatsumsatz

Daneben werden eine Reihe von Variablen benutzt, deren Bedeutung beim Lesen des Listings leicht klar wird, wenn es erforderlich ist.

Hier nun das Listing des Programms Busyman.

```
  1 BORDER 0: PAPER 0: INK 6: CLS : REM ? pcb
 10 LET m=9000: GO TO m
 15 INPUT "Datum = ENTER=";(d$);" "' LINE m$
 16 IF m$<>"" THEN  LET d$=m$
 17 RETURN
 20 LET anz=20: GO SUB 75
 30 LET e$="JanFebMarAprMaiJunJulAugSepOktNovDez": LET e$=e$+e$
 40 LET em=1: DIM j(12,6): DIM w$(1,12)
 50 LET wg=1: DIM o(wg): LET o(1)=1
 60 DIM u(wg,3): LET max=100: LET mwst=.14: LET hu=100: LET xx=10000: LET m1=1+
mwst
 62 GO SUB 70
 65 LET stkj=0: LET eks=0: LET vks=0: LET eksj=0: LET vksj=0
 67 GO TO m
 70 DIM b$(max,2): DIM e(max): DIM z(max): DIM y$(max): RETURN
 75 DIM g$(anz,2): DIM v(anz): RETURN
 80 PRINT AT 21,0;"******* FALSCHE EINGABE! *******"
 82 FOR i=1 TO 10: BEEP .05,i: NEXT i
 84 PRINT AT 21,0;,,: RETURN
 90 PRINT ' INK 5;"Art.Nr EK      Stk-verk      Rest": RETURN
 95 PRINT ''TAB 8;"ENDE DER LISTE": RETURN
 99 CLS : PRINT AT 9,9;" BITTE WARTEN ": RETURN
100 LET f=0: IF m$="" THEN  GO TO 108: REM wg
104 FOR l=1 TO wg: IF VAL m$=l THEN  RETURN
106 NEXT l
108 LET f=1: RETURN
110 LET f=0: IF m$="" THEN  GO TO 118: REM Artnr
111 FOR l=1 TO LEN m$
112 IF CODE m$(l)<48 OR CODE m$>57 THEN  GO TO 118
113 NEXT l
114 IF VAL m$>=1 AND VAL m$<=250 THEN  RETURN
118 LET f=1: RETURN
120 LET f=0: LET su1=0: REM ek/vk
121 IF m$="" THEN  GO TO 128
123 FOR l=1 TO LEN m$
124 IF su1=0 AND m$(l)="." THEN  LET su1=1: GO TO 126
125 IF CODE m$(l)<48 OR CODE m$(l)>57 THEN  GO TO 128
126 NEXT l
127 IF su1<=1 THEN  RETURN
128 LET f=1: RETURN
130 LET f=0: IF m$="" THEN  GO TO 138: REM stk
134 FOR l=1 TO LEN m$
```

```
135 IF CODE m$(l)<48 OR CODE m$(l)>57 THEN  GO TO 138
136 NEXT l
137 RETURN
138 LET f=1: RETURN
140 GO SUB 130: IF f=1 THEN  GO TO 5100
141 IF VAL m$>750 OR VAL m$<1 THEN  GO TO 5100
143 LET max=VAL m$: GO SUB 70
144 GO TO 5100
145 GO SUB 130: IF f=1 THEN  GO TO 5100
146 IF VAL m$>150 OR VAL m$<1 THEN  GO TO 5100
148 LET anz=VAL m$: GO SUB 75
149 GO TO 5100
150 INPUT "Eintrag loeschen? J/N ";m$: IF m$<>"j" THEN  GO TO 5000
154 LET b$(i)="  ": LET z(i)=0: LET e(i)=0: LET y$(i)="  "
156 GO SUB 99
158 FOR k=i TO max-1
160 LET b$(k)=b$(k+1): LET z(k)=z(k+1): LET e(k)=e(k+1): LET y$(k)=y$(k+1)
162 IF b$(k,1)=" " THEN  GO TO 5000
164 NEXT k
166 LET b$(max)="  ": LET z(max)=0: LET e(max)=0: LET y$(max)="  "
167 GO SUB 350
168 GO TO 5000
170 GO SUB 120: IF f=1 THEN  GO TO 5100
172 LET mwst=VAL m$/100: LET m1=1+mwst: GO TO 5100
180 INPUT "Erster Monat Geschjahr(Jan=1)"; LINE a$: GO SUB 130
184 FOR l=1 TO LEN a$
186 IF CODE a$(l)<48 OR CODE a$(l)>57 THEN  GO TO 182
188 NEXT l
190 LET em=VAL a$
192 IF em<1 OR em>12 THEN  GO TO 182
194 RETURN
200 IF INKEY$<>"" THEN  GO TO 200: REM korrektur
201 POKE 23689,PEEK 23689+1
202 PRINT ,,
203 POKE 23689,PEEK 23689+1
204 RETURN
250 LET per=0: REM ordnen
252 FOR l=1 TO max-1: IF b$(l+1,1)=" " THEN  GO TO 260
253 IF b$(l)>b$(l+1) THEN  LET per=per+1
254 NEXT l
260 IF per=0 THEN  GO TO 285
264 FOR s=l TO 2 STEP -1
```

```
268 FOR k=1 TO s-1
270 IF b$(s)<b$(k) THEN  GO TO 291
272 NEXT k
280 NEXT s
285 GO SUB 350
290 GO TO 5000
291 LET m$=b$(s): LET pe=e(s): LET pz=z(s): LET x$=y$(s)
292 FOR j=s TO k+1 STEP -1
293 LET b$(j)=b$(j-1): LET e(j)=e(j-1): LET z(j)=z(j-1): LET y$(j)=y$(j-1)
294 NEXT j
295 LET b$(k)=m$: LET e(k)=pe: LET z(k)=pz: LET y$(k)=x$
296 GO TO 250
300 CLS : PRINT  INK 5;"WARENGRUPPEN:": FOR i=1 TO wg: PRINT  INK 4;i;" ";w$(i)
,: NEXT i: RETURN
350 REM ordung
352 LET k=2
355 FOR i=1 TO wg: LET o(i)=1: NEXT i
360 FOR i=2 TO max
365 IF b$(i,1)=" " THEN  GO TO 390
370 IF b$(i,1)>b$(i-1,1) THEN  LET o(k)=i: LET k=k+CODE b$(i,1)-CODE b$(i-1,1)
375 IF k>wg THEN  GO TO 390
380 NEXT i
392 FOR i=2 TO wg: IF o(i)<o(i-1) THEN  LET o(i)=o(i-1)
394 NEXT i
399 RETURN
400 DEF FN r(a,b,c)=INT (100*(a/b-c))/100: DEF FN k(a,b,c)=INT (10000*a*b/c)/10
0: DEF FN l(a,b)=INT (10000*a/b)/100
500 CLS : PRINT "          Problem          "
510 LET a1=i
525 PRINT ' INK 5;"Artikel nicht im Lager"'' INK 2;"    WG        Art.Nr    VK"
530 PRINT : PRINT i;" ";w$(t);" ";CODE g$(i,2);"  DM ";v(i)
550 PRINT '''"1 = Quittung neu eingeben","2 = Quittung loeschen","3 = Artikel i
n Lager aufnehmen","4 = Menu"
560 PAUSE 0: LET m$=INKEY$: IF m$="4" THEN  GO TO m
570 IF m$="1" THEN  GO TO 600
572 IF m$="2" THEN  GO TO 700
574 IF m$="3" THEN  GO TO 800
578 GO TO 560
600 PRINT "********Neue Eingabe************"'
610 INPUT "WG =";m$: GO SUB 100
615 IF f=1 THEN  GO TO 610
620 PRINT m$;"  ";: LET g$(i,1)=CHR$ (VAL m$)
```

```
622 INPUT "Art.Nr = ";m$: GO SUB 110
624 IF f=1 THEN  GO TO 622
628 LET g$(i,2)=CHR$ (VAL m$): PRINT m$;
640 INPUT "VK = ";m$: GO SUB 120
645 IF f=1 THEN  GO TO 640
648 LET v(i)=VAL m$: PRINT "DM ";v(i)
680 PAUSE 50: GO TO 1360
700 PRINT '"Bitte warten"
720 FOR k=i TO anz-1: LET g$(k)=g$(k+1): LET v(k)=v(k+1): NEXT k
740 LET g$(anz)="  ": LET v(anz)=0: GO TO 1360
800 FOR k=o(wg) TO max
820 IF b$(k,1)=" " THEN  GO TO 850
830 NEXT k
840 GO TO 999
850 LET b$(k)=g$(i)
860 INPUT "EK = ";m$: GO SUB 120
864 IF f=1 THEN  GO TO 860
866 LET e(k)=VAL m$
870 INPUT "Anzahl vor Verkauf = ";m$: GO SUB 130
874 IF f=1 THEN  GO TO 870
876 LET z(k)=VAL m$
899 GO TO 1360
900 LET x$=CHR$ wag+CHR$ fn
920 FOR l=o(wag) TO max
922 IF b$(l,1)=" " THEN  GO TO 935
925 IF b$(l)=x$(1 TO 2) THEN  GO TO 950
930 NEXT l
935 PRINT AT 21,8;"Nicht vorhanden": PAUSE 150: CLS : GO SUB 300: GO TO 1110
950 IF z(l)<1 THEN  GO TO 935
960 GO TO 1120
999 CLS : PRINT "Kein Platz vorhanden": PAUSE 200: GO TO m
1000 CLS : PRINT  INK 3;TAB 7;"Tagesgeschaefte"'' INK 5;;d$: PRINT ''''"    1 = Ve
rkaeufe eingeben"''"    2 = Liste der Verkaeufe"''"    3 = Tagesabrechnung"''"    4
 = Menu": PAUSE 0
1010 LET u$="0"
1030 LET m$=INKEY$
1032 IF m$="1" THEN  GO TO 1100
1034 IF m$="2" THEN  GO TO 1200
1036 IF m$="3" THEN  GO TO 1300
1038 IF m$="4" THEN  GO TO 9000
1039 GO TO 1030
1100 CLS : IF g$(1,1)=" " THEN  GO TO 1104
```

```
1102 INPUT "Neuer Tag? J/N "; LINE m$
1103 IF m$="j" THEN  DIM g$(anz,2): DIM v(anz)
1104 FOR i=1 TO anz: IF g$(i,1)=" " THEN  GO TO 1106
1105 NEXT i
1106 LET n=i: GO SUB 300
1110 PRINT '"EINGABE                  Verkaeufe "; INK 5;d$
1112 PRINT AT wg/2+4,0; INK 7;"W=Wiederhole E=Ende  K=Korrektur"
1114 PRINT  INK 3;"Warengr.  Art.Nr    Verk.Preis  "
1116 PRINT AT wg/2+6,0;
1119 LET su1=0: LET i=0: LET t$=""
1120 PRINT " ";CHR$ 8;
1122 PAUSE 0: LET m$=INKEY$: BEEP .01,1
1124 IF CODE m$=13 THEN  GO TO 1150
1125 IF m$="w" THEN  PRINT wag;TAB 11;fn;TAB 20;vkp: GO TO 1186
1126 IF m$="k" THEN  GO TO 1104
1127 IF m$="e" THEN  GO TO 1000
1128 IF i=2 AND m$="." THEN  GO TO 1130
1129 IF CODE m$>57 OR CODE m$<48 THEN  GO TO 1121
1130 PRINT m$;: LET t$=t$+m$
1140 IF INKEY$<>"" THEN  GO TO 1140
1145 GO TO 1120
1150 IF t$="" THEN  GO TO 1104
1151 LET i=i+1: IF i>1 THEN  GO TO 1160
1154 FOR k=1 TO wg: IF VAL t$=k THEN  GO TO 1158
1155 NEXT k
1156 GO SUB 80: GO TO 1104
1158 LET wag=VAL t$: LET t$=""
1159 PRINT TAB 11;: GO TO 1120
1160 IF i>2 THEN  GO TO 1170
1162 LET m$=t$
1164 GO SUB 110: IF f=1 THEN  GO SUB 80: GO TO 1104
1166 LET fn=VAL t$: LET t$=""
1168 PRINT TAB 20;: GO TO 900
1169 GO TO 1120
1170 REM Preis
1172 LET m$=t$: GO SUB 120
1174 IF f=1 THEN  GO SUB 80: GO TO 1104
1176 LET vkp=VAL t$
1178 PRINT " ": LET t$=""
1186 LET g$(n,1)=CHR$ wag
1188 LET g$(n,2)=CHR$ fn
1190 LET v(n)=vkp
```

```
1191 LET n=n+1
1192 IF n>anz THEN  CLS : PRINT  FLASH 1;"Vkliste voll, bitte abrechnen!": PAUSE
 100: GO TO 1000
1193 IF INT (n/10)>=n/10 THEN  GO TO 1104
1195 GO TO 1119
1200 CLS : LET su1=0: PRINT  INK 3;TAB 5;"Liste der Verkaeufe"' INK 5;d$
1220 FOR i=1 TO anz
1230 IF g$(i,1)=" " THEN  GO TO 1270
1240 PRINT i;TAB 3;w$(CODE g$(i,1));TAB 16;CODE g$(i,2);TAB 22;v(i);TAB 28;"DM"
1250 LET su1=su1+v(i)
1260 NEXT i
1270 PRINT ' INK 4;"Gesamtverkauf = ";su1;" DM"
1280 PAUSE 0: GO TO 1000
1300 CLS : IF u$="j" THEN  GO TO 1310
1305 LET a1=1: LET u$="0"
1310 PRINT  INK 3;" T a g e s u m s a t z ", INK 5;d$
1320 PRINT  INK 4;"Warengruppe Anz Vk      Ek"
1330 FOR i=1 TO wg: PRINT AT i+2,0;w$(i): NEXT i
1340 LET a2=anz
1350 DIM r(wg,3)
1360 FOR i=a1 TO a2
1361 IF g$(i,1)=" " THEN  GO TO 1380
1362 LET t=CODE g$(i,1)
1365 LET r(t,1)=r(t,1)+1: LET r(t,3)=r(t,3)+v(i)
1367 GO SUB 1500
1368 IF l>max THEN  LET l=max
1369 IF u$<>"j" THEN  GO TO 1375
1370 LET u(t,1)=u(t,1)+1: LET u(t,3)=u(t,3)+v(i)
1375 NEXT i
1380 LET p=0: LET su1=0: LET su2=0
1385 FOR i=1 TO wg
1386 PRINT AT i+2,13;r(i,1);TAB 16;r(i,3);TAB 24;r(i,2)
1387 LET p=p+r(i,1): LET su1=su1+r(i,2): LET su2=su2+r(i,3)
1388 NEXT i
1389 PRINT  INK 2;" S U M M E = ";TAB 13;p;TAB 16;su2;TAB 24;su1
1390 PRINT "Kalkulation:": IF su1>0 AND su2>0 THEN  LET re=FN r(su2,m1,su1): PRI
NT "RE=DM ";re'"Kalk = ";FN k(re,m1,su2);"/";FN l(re,su1)
1395 PRINT "Laufender Monat"''"VK=";vks;" DM";TAB 16;"EK=";eks;" DM": IF eks>0 A
ND vks>0 THEN  LET re=FN r(vks,m1,eks): PRINT "RE = DM ";re'"Kalk = ";FN k(re,m1
,vks);"/";FN l(re,eks)
1400 IF u$="j" THEN  DIM g$(anz,2): DIM v(anz): PAUSE 0: GO TO 1000
1470 IF u$="0" THEN  INPUT "Bestand updaten? J/N"; LINE u$: IF u$="j" THEN  CLS
```

```
: GO TO 1310
1490 GO TO 1000
1500 REM EK suchen
1505 LET su1=CODE g$(i,1)
1510 FOR l=o(su1) TO max
1515 IF b$(l,1)=" " THEN  GO TO 500
1520 IF g$(i)<>b$(l) THEN  NEXT l
1535 IF z(l)<=0 THEN  GO TO 1580
1540 LET r(t,2)=r(t,2)+e(l)
1545 IF u$<>"j" THEN  GO TO 1570
1550 LET y$(l)=CHR$ (CODE y$(l)+1)
1555 LET u(t,2)=u(t,2)+e(l)
1560 LET z(l)=z(l)-1
1565 LET eks=eks+e(l): LET vks=vks+v(i)
1566 LET stkj=stkj+1: LET eksj=eksj+e(l): LET vksj=vksj+v(i)
1570 RETURN
1580 NEXT l
1590 GO TO 500
1700 FOR k=1 TO max-1
1705 IF b$(k,1)=" " THEN  RETURN
1710 IF z(k)>0 THEN  GO TO 1750
1720 FOR l=k TO max-1
1730 LET y$(l)=y$(l+1): LET b$(l)=b$(l+1): LET e(l)=e(l+1): LET z(l)=z(l+1)
1735 IF b$(l,1)=" " THEN  GO TO 1745
1740 NEXT l
1745 LET b$(max)="  ": LET k=k-1
1750 NEXT k
1760 RETURN
2000 CLS : PRINT  INK 3;" U M S A T Z  und B E S T A N D "; INK 5;d$''
2010 PRINT "  1 = Bestandsuebersicht"'''  2 = Monatsuebersicht"'''  3 = Monat im
 Detail"'''  4 = Jahresueberblick"'''  5 = Monatsabschluss/anfang"'''  6 = Menu"
2020 PAUSE 0: LET m$=INKEY$: IF m$="6" THEN  GO TO m
2026 IF m$="1" THEN  GO TO 5800
2028 IF m$="5" THEN  GO TO 2900
2032 IF m$="2" THEN  GO TO 2100
2034 IF m$="3" THEN  GO TO 2400
2036 IF m$="4" THEN  GO TO 2600
2040 GO SUB 80: GO TO 2000
2100 CLS : PRINT  INK 3;"   M O N A T S - U M S A T Z    "' INK 6;d$
2110 PRINT  INK 5;"WG              Stk VK     EK     ": LET p=0: LET su1=0: LET su
2=0
2150 FOR i=1 TO wg
```

```
2160 PRINT w$(i),;u(i,1);TAB 19;u(i,3);TAB 26;u(i,2)
2180 IF u(i,2)<=0 OR u(i,3)<=0 THEN  GO TO 2195
2190 LET re=FN r(u(i,3),m1,u(i,2)): PRINT "RE=DM ";re'"Kalk: ";FN k(re,m1,u(i,3)
);"/";FN l(re,u(i,2))''
2200 NEXT i
2220 PRINT '' INK 3;"S U M M E = ";TAB 16;p;TAB 19;su2;TAB 26;su1
2230 PRINT '' INK 3;"Laufender Monat:"'' INK 4;"VK = DM ";vks;TAB 16;"EK = DM ";
eks: IF eks>0 AND vks>0 THEN  LET re=FN r(vks,m1,eks): PRINT "RE=DM ";re'"Kalk:"
;FN k(re,m1,vks);"/";FN l(re,eks)
2240 GO SUB 95
2300 PAUSE 0: GO TO 2000
2400 CLS : PRINT TAB 5;"Monatsumsatz im Detail"' INK 5;d$'
2435 LET l=1
2440 FOR i=1 TO max
2450 IF b$(i,1)=" " THEN  GO TO 2480
2454 IF i=o(l) AND i>1 THEN  INPUT  INK 4;"ENTER = weiter "; LINE m$: CLS
2455 IF i=o(l) THEN  PRINT ''; INK 4;l;" ";w$(l);: GO SUB 90: IF l<wg THEN  LET
l=l+1: GO TO 2455
2460 PRINT CODE b$(i,2);TAB 6;e(i);TAB 15;CODE y$(i)-32;TAB 27;z(i)
2470 NEXT i
2480 GO SUB 95
2490 PAUSE 0: GO TO 2000
2600 CLS : PRINT TAB 8;"Jahresueberblick"' INK 5;d$''
2610 FOR i=1 TO 12
2615 IF j(i,2)=0 OR j(i,3)=0 THEN  GO TO 2650
2620 PRINT '' INK 2;"Monat  Stk    Vk        Ek"
2625 PRINT  INVERSE 1;e$(3*(em-1+i)-2 TO 3*(em-1+i)); INVERSE 0;TAB 8;j(i,1);TAB
 14;j(i,3);TAB 24;j(i,2)
2630 LET re=FN r(j(i,3),m1,j(i,2))
2635 PRINT ;"RE = ";re;TAB 15;"Kalk ";FN k(re,m1,j(i,3));"/";FN l(re,j(i,2))
2640 PRINT '"Staffel";TAB 8;j(i,4);TAB 14;j(i,6);TAB 24;j(i,5)
2645 LET re=FN r(j(i,6),m1,j(i,5)): PRINT "RE = ";re;TAB 15;"Kalk ";FN k(re,m1,j
(i,6));"/";FN l(re,j(i,5))
2650 NEXT i
2655 PRINT '' INK 2;"Laufendes Jahr:"; INK 4'"VK= DM ";vksj;TAB 16;"EK= DM ";eks
j
2660 IF eksj>0 AND vksj>0 THEN  LET re=FN r(vksj,m1,eksj): PRINT "RE= DM ";re: P
RINT "Kalk = ";FN k(re,m1,vksj);"/";FN l(re,eksj)
2665 GO SUB 95
2670 PAUSE 0: GO TO 2000
2900 CLS : INPUT "Alle Daten eingegeben? J/N "; LINE m$: IF m$<>"j" THEN  GO TO
m
```

```
2920 PRINT "Bitte warten"
2921 FOR k=1 TO 12
2922 IF j(k,1)=0 THEN  GO TO 2925
2923 NEXT k
2925 LET j(k,1)=p: LET j(k,2)=eks: LET j(k,3)=vks
2926 LET j(k,4)=stkj: LET j(k,5)=eksj: LET j(k,6)=vksj
2930 DIM u(wg,3): DIM y$(max)
2955 LET eks=0: LET vks=0: DIM g$(anz,2): DIM v(anz)
2960 GO SUB 1700
2965 GO SUB 350
2970 INPUT "Ende Geschaeftsjahr? J/N "; LINE a$
2975 IF a$<>"j" THEN  GO TO 2000
2976 INPUT "Bitte ende eingeben"; LINE m$
2977 LET a$="0"
2978 IF m$="ende" THEN  LET a$="j"
2979 CLS : PRINT AT 10,13; FLASH 1;"Ende": PAUSE 150
2980 IF a$="j" THEN  LET eksj=0: LET vksj=0: LET stkj=0
2985 IF a$="j" THEN  DIM j(12,6)
2999 GO TO 2000
5000 CLS : PRINT  INK 3;"  Eingabe/Aenderung der Daten  ": PRINT '' INK 6;"1 =
Veraenderung Bestand"'''"2 = Bestand eingeben"'''"3 = WG aufsetzen"'''"4 = Bestand
ordnen"'''"5 = Menu"
5020 PAUSE 0: LET m$=INKEY$: IF m$="5" THEN  GO TO m
5050 IF m$="4" THEN  CLS : PRINT  FLASH 1;"Bitte warten": GO TO 250
5054 IF m$="1" THEN  GO TO 5600
5056 IF m$="3" THEN  GO TO 5100
5058 IF m$="2" THEN  GO TO 5300
5060 GO SUB 80: GO TO 5000
5100 CLS : PRINT "WG eingeben"''
5124 PRINT '"1 = Anzahl der Warengruppen";TAB 28;wg
5126 PRINT '"2 = Namen der Warengruppen"
5127 PRINT '"3 = Beginn Geschaeftsjahr = ";e$(3*em-2 TO 3*em)
5128 PRINT '"4 = Mehrwertsteuer = ";100*mwst;"%"'''"5 = Anzahl Warentypen = ";max
5129 PRINT '"6 = Maximale Zahl Tagesvk = ";anz'''"7 = Menu"
5130 PAUSE 0: LET m$=INKEY$: IF m$="7" THEN  GO TO 5000
5150 IF m$="4" THEN  INPUT "Wieviel % Mwst? ";m$: GO TO 170
5154 IF m$="5" THEN  INPUT "Anzahl Warentypen?(1-750) ";m$: GO TO 140
5156 IF m$="6" THEN  INPUT "Anzahl Vk pro Tag?(1-150) ";m$: GO TO 145
5160 IF m$="3" THEN  GO SUB 180: GO TO 5100
5180 IF m$="2" THEN  GO TO 5220
5190 IF m$="1" THEN  GO TO 5200
5195 GO TO 5130
```

```
5200 INPUT "Anzahl der WG (1-20)= ";m$
5210 GO SUB 130: IF f=1 THEN  GO TO 5200
5212 LET wg=VAL m$: IF wg<1 OR wg>20 THEN  GO TO 5200
5213 DIM o(wg): DIM w$(wg,12): DIM u(wg,3)
5214 FOR i=1 TO wg: LET o(i)=1: NEXT i
5215 PRINT AT 3,28;"  ";AT 3,28;wg
5216 GO TO 5100
5220 IF wg<=0 THEN  PRINT '"Anzahl WG = 0": PAUSE 150: GO TO 5100
5222 CLS : PRINT
5224 FOR i=1 TO wg
5226 PRINT : PRINT "WG ";i;" = ";TAB 8;w$(i);: INPUT  LINE m$;: IF m$<>"" THEN
LET w$(i)=m$
5227 POKE 23689,PEEK 23689+1: PRINT  PAPER 1;i;TAB 8;w$(i)
5228 NEXT i
5290 PAUSE 0: GO TO 5000
5300 CLS
5303 FOR s=o(wg) TO max
5304 IF b$(s,1)<>" " THEN  GO TO 5306
5305 GO TO 5310
5306 NEXT s
5308 GO TO 999
5310 GO SUB 300: PRINT : PRINT  INK 3;"     B E S T A N D   eingeben  "
5330 INPUT "Nummer der WG (0=Ende) = "; LINE m$
5332 IF m$="0" THEN  GO TO 250
5333 GO SUB 100
5334 IF f=1 THEN  GO SUB 80: GO TO 5310
5336 LET i=VAL m$
5400 CLS : PRINT "Eingabe   "; INK 4;w$(i)
5402 PRINT  INK 5;"Artikel-Nr   Ekpreis      Stueck"
5405 INPUT "Artikel-Nr (0=Ende)= "; LINE m$: IF m$="0" THEN  GO TO 5300
5406 GO SUB 110
5407 IF f=1 THEN  GO SUB 80: GO TO 5405
5410 LET fn=VAL m$: PRINT fn;
5415 LET m$=CHR$ i+CHR$ fn
5420 FOR l=o(i) TO max: IF b$(l,1)=" " THEN  GO TO 5430
5421 IF b$(l)=m$ THEN  GO TO 5425
5422 NEXT l
5424 GO TO 5430
5425 PRINT " vorhanden";TAB 14;e(l): PAUSE 250: GO TO 5400
5430 INPUT "Ekpreis = "; LINE m$: GO SUB 120
5432 IF f=1 THEN  GO SUB 80: GO TO 5430
5436 LET ek=VAL m$: PRINT TAB 14;ek;
```

```
5440 INPUT "Anzahl = "; LINE m$: GO SUB 120
5443 IF f=1 THEN  GO TO 5440
5446 LET az=VAL m$: PRINT TAB 26;az
5460 INPUT "Eintrag ok?J/N "; LINE m$
5465 IF m$="j" THEN  GO TO 5500
5470 GO SUB 201
5480 GO TO 5405
5500 LET b$(s,1)=CHR$ i
5502 LET b$(s,2)=CHR$ fn
5508 LET e(s)=ek
5510 LET z(s)=az
5520 LET s=s+1
5530 IF s>max THEN  PRINT "Kein Platz fuer weiteren Eintrag": PAUSE 150: GO TO m
5550 GO TO 5405
5600 GO SUB 300
5610 PRINT AT 14,0;"1 = Nachlieferung"'"2 = Retoursendung"'"3 = Umtausch"'"4 = K
orrektur einer Eingabe"'"5 = Menu"
5615 PAUSE 0: LET m$=INKEY$: IF m$="5" THEN  GO TO 5000
5616 IF m$>"4" OR m$<="0" THEN  GO SUB 80: GO TO 5615
5617 LET a$=m$
5620 IF m$="1" THEN  PRINT AT 14,4;"Nachlieferung"
5621 IF m$="2" THEN  PRINT AT 15,4;"Retoursendung"
5622 IF a$="3" THEN  PRINT AT 16,4;"Umtausch"
5623 IF a$="4" THEN  PRINT AT 17,4;"Korrektur einer Eingabe"
5650 INPUT "WG = "; LINE m$: GO SUB 100
5651 IF f=1 THEN  GO SUB 80: GO TO 5650
5652 LET w=VAL m$
5658 PRINT AT 19,0; INK 5;w$(w); INK 6;TAB 17;" ";
5660 INPUT "Art.Nr = "; LINE m$: GO SUB 110
5662 IF f=1 THEN  GO SUB 80: GO TO 5660
5668 LET fn=VAL m$: PRINT fn
5690 FOR i=o(w) TO max
5695 IF b$(i,1)=" " THEN  GO TO 5725
5700 IF CODE b$(i,1)<>w THEN  GO TO 5720
5702 IF CODE b$(i,2)<>fn THEN  GO TO 5720
5710 GO TO 5730
5720 NEXT i
5725 PRINT "Nicht gefunden": PAUSE 150: GO TO 5000
5730 PRINT AT 20,0; INK 3;"Ek:";e(i);" DM";TAB 17;"Bestand:";z(i)
5750 IF a$="1" THEN  INPUT "Anzahl = ";su1: LET z(i)=z(i)+su1
5755 IF a$="2" THEN  INPUT "Anzahl = ";su1: LET z(i)=z(i)-su1
5758 IF a$="3" THEN  LET z(i)=z(i)+1
```

```
5759 IF a$="4" THEN  GO TO 5778
5760 PRINT TAB 11; INK 2;"Neuer Bestand:";z(i)
5770 INPUT "M = Menu ,Enter = weiter"; LINE m$
5775 IF m$="m" THEN  GO TO 5000
5776 GO TO 5600
5778 PRINT AT 21,0,,: INPUT "(Enter=OK,L=Loeschen) WG="; LINE m$: IF m$="" THEN
 GO TO 5783
5779 IF m$="l" THEN  GO TO 150
5781 GO SUB 100: IF f=1 THEN  GO SUB 80: GO TO 5780
5782 LET b$(i,1)=CHR$ (VAL m$)
5783 PRINT AT 19,0; PAPER 3;w$(CODE b$(i,1)): INPUT "Neue Art.Nr = (Enter=Ok) ";
 LINE m$: IF m$="" THEN  GO TO 5786
5784 GO SUB 110: IF f=1 THEN  GO SUB 80: GO TO 5783
5785 LET b$(i,2)=CHR$ (VAL m$)
5786 PRINT AT 19,18;"      ";AT 19,18; PAPER 3;CODE b$(i,2)
5790 INPUT "EK (ENTER=ok) "; LINE m$: IF m$="" THEN  GO TO 5794
5792 GO SUB 120: IF f=1 THEN  GO SUB 80: GO TO 5790
5793 LET e(i)=VAL m$
5794 PRINT AT 20,3;"           ";AT 20,3; PAPER 3;e(i);" DM": INPUT "Neue Stkzahl
 = (Enter=Ok) "; LINE m$: IF m$="" THEN  GO TO 5797
5795 GO SUB 130: IF f=1 THEN  GO SUB 80: GO TO 5794
5796 LET z(i)=VAL m$
5797 PRINT AT 20,25; PAPER 3;z(i)
5799 GO TO 250
5800 CLS : PRINT  INK 3;"      Bestandsuebersicht     "''' INK 4;TAB 7;"1 = Ueber
sicht"''TAB 7;"2 = Detail WG"''TAB 7;"3 = Menu"
5810 PAUSE 0: LET m$=INKEY$: IF m$="3" THEN  GO TO 2000
5811 IF m$<>"1" AND m$<>"2" THEN  GO TO 5810
5812 IF m$="1" THEN  CLS : GO TO 5818
5814 GO SUB 300: INPUT "WG (0 = alle) = ";su3: IF su3>wg THEN  GO SUB 80: GO TO
5814
5815 LET w=su3
5816 GO TO 5900
5818 DIM r(wg,2): PRINT  INK 3;TAB 7;"BESTANDS-UEBERSICHT"' INK 6;d$; INK 2'''"Be
stand                  Wert"
5820 FOR i=1 TO max
5822 IF b$(i,1)=" " THEN  GO TO 5840
5824 LET k=CODE b$(i,1)
5826 LET r(k,1)=r(k,1)+z(i): LET r(k,2)=r(k,2)+z(i)*e(i)
5830 NEXT i
5840 LET su1=0: LET su2=0
5850 FOR i=1 TO wg: PRINT w$(i);TAB 15;r(i,1);TAB 21;r(i,2): LET su1=su1+r(i,1):
```

```
 LET su2=su2+r(i,2): NEXT i
5860 PRINT '' INK 3;"S U M M E  =";TAB 15;su1;TAB 21;su2
5870 PAUSE 0: GO TO 2000
5900 IF w=0 THEN  FOR w=1 TO wg
5902 CLS : PRINT  INK 6;;d$; INK 3'"Bestand ";w$(w)' INK 5;"Artikel-Nr     Ekpre
is   Anzahl"
5905 LET su1=0: LET su2=0
5910 FOR i=o(w) TO max
5915 IF b$(i,1)=" " THEN  GO TO 5932
5920 IF CODE b$(i,1)=w THEN  PRINT CODE b$(i,2);TAB 15;e(i);TAB 26;z(i): LET su1
=su1+e(i)*z(i): LET su2=su2+z(i)
5930 NEXT i
5932 PRINT ' INK 3;"S U M M E  DM ";su1;TAB 26;su2
5960 INPUT  INK 2;"ENTER=weiter"; LINE m$: IF su3=0 THEN  NEXT w
5970 GO TO 2000
6000 CLS : PRINT AT 9,0;"Programm / Daten sichern? J/N": PAUSE 0
6040 IF INKEY$="n" THEN  LOAD *"m";1;"run"
6050 CLS : PRINT "Daten sichern";AT 5,0;"1 = Microdrive"''"2 = Cassette"''"3 = M
enu": PAUSE 0: LET m$=INKEY$
6060 IF m$="1" THEN  GO TO 6200
6070 IF m$="2" THEN  CLS : GO TO 6100
6080 IF m$="3" THEN  GO TO 9000
6100 PRINT AT 9,0;"Bitte Cassette BUSYMAN einlegen,auf Aufnahme druecken und  EN
TERtippen"
6110 SAVE "Busyman" LINE 1
6120 PRINT "Bitte Band zum VERIFY zurueck-  spulen und Play druecken": VERIFY ""
6190 GO TO 6050
6200 PRINT '''"Bitte warten"
6220 ERASE "m";1;"Busyman"
6230 SAVE *"m";1;"Busyman" LINE 1
6240 VERIFY *"m";1;"Busyman"
6290 GO TO 6050
9000 CLS : PRINT  INK 3;"  Lager/Umsatzverwaltung-Menu"''"***********************
*********";AT 2,0; INK 5;d$;TAB 27;"? pcb"
9030 PRINT ''''  1  = Tagesgeschaeft""''"  2  = Bestands/Umsatzuebersicht"''''
3  = Eingabe/Aenderung Daten"''''  4  = Datum"''''  5  = Ende"
9100 PAUSE 0: LET m$=INKEY$: IF m$="4" THEN  GO SUB 15: GO TO m
9116 IF m$="1" THEN  GO TO 1000
9117 IF m$="2" THEN  GO TO 2000
9118 IF m$="3" THEN  GO TO 5000
9119 IF m$="5" THEN  GO TO 6000
9120 GO SUB 80: GO TO 9040
```

Um zunächst den Speicherplatz für alle Variablen zu reservieren, muß nach dem Sichern des Programms, was wieder mit der LINE 1 Option geschieht, ein GO TO 20 eingegeben werden. Anschließend geht es mit dem Punkt 3 in das Untermenü zur Eingabe/Änderung von Daten. Hier muß wiederum der Punkt 3 gewählt werden: Aufsetzen der Warengruppen.

Nun wird die Anzahl der verschiedenen Warengruppen eingegeben. Hier sollte man eher eine Gruppe zuviel als zu wenig wählen, da diese Zahl später nicht mehr korrigiert werden kann, ohne Daten zu verlieren. Dann kann jeder Warengruppe ein Name zugeordnet werden, der maximal 12 Zeichen lang sein darf. Diese Namen können später jederzeit geändert werden.

Die Anzahl der Warentypen ist ebenfalls festzulegen. Der 48K Spectrum erlaubt mit dem Programm bis zu 750 Artikeltypen gleichzeitig zu verwalten. Sollte man in jedem Fall wesentlich weniger Artikel haben, empfiehlt es sich, eine kleinere Zahl einzugeben, da einige Operationen im Programm dann schneller ablaufen. Die Anzahl der Verkäufe pro Tag, die maximal mit einer Abrechnung verarbeitet werden können, muß ebenfalls eingegeben werden. Sie sollte 150 nicht überschreiten. Dies gilt allerdings nur für die hier gezeigte Version des Programms. Natürlich können Sie diese Zahl auch viel größer wählen, wenn der Speicherplatz dies erlaubt, d. h., Sie zum Beispiel wesentlich weniger als 750 Artikeltypen verwalten wollen. Sie müssen dann nur in der Zeile 146 anstelle der 150 die entsprechende Zahl eingeben.

Da nicht jeder Geschäftsmann sein Geschäftsjahr im Januar beginnt, kann auch dieser Monat festgelegt werden. In der Jahresübersicht werden dann die abgeschlossenen Monate

den richtigen Monatsnamen zugeordnet. Natürlich läßt sich auch der Mehrwertsteuersatz, der mit 14% vorgegeben ist, jederzeit verändern.

Nachdem diese Arbeiten erledigt sind, kann wieder ins Hauptmenü zurückgekehrt werden. Nun sollte noch das Datum mit der Option 4 eingegeben werden. Dann können Sie aber im Prinzip den Warenbestand eingeben.

Dazu wird auch der Punkt 3 des Hauptmenüs angewählt. Im erscheinenden Untermenü wird nun über den Punkt 2 (Bestand eingeben) diese Routine aufgerufen. Zunächst wird die Nummer der Warengruppe, die bearbeitet werden soll und dann Artikelnummer, Einkaufspreis und Stückzahl verlangt. Jede Eingabe muß bestätigt werden. Natürlich lassen sich falsche Eingaben mit der Option 'Änderung Daten' durchführen.

Wenn der Bestand aller Warengruppen eingegeben ist, läßt sich bereits ein Überblick über das Lager durchführen. Dies erreicht man durch die zweite Option im Hauptmenü, die in das Untermenü 'Umsatz und Bestand' führt. Die Bestandsübersicht kann einmal für jede Warengruppe im Detail gesehen werden, zum zweiten für die gesamten Warengruppen. Die Monatsübersicht gibt für jeden Artikel die Verkaufszahlen im laufenden Monat an wie auch den Restbestand (Punkt 3), oder den Überblick für die Warengruppen (Punkt 2).

Am Ende eines jeden Monats muß der Monatsabschluß durchgeführt werden. Damit werden Gesamtertrag und Umsatz gespeichert (diese Zahlen können im Jahresüberblick abgerufen werden) und die Verkaufszahlen des laufenden Monats wieder auf Null gesetzt.

Das Tagesgeschäft ist auch einfach zu erledigen. Aus dem Hauptmenü wählt man den Punkt 1 und gelangt wieder in ein Untermenü. Die Option 1 bringt den Benutzer zur Routine, welche die Eingabe der Verkäufe handhabt. Es werden Warengruppe, Artikelnummer und erzielter Verkaufspreis

eingegeben. Durch Drücken von 'K' kann eine Korrektur der Eingabe durchgeführt werden, mit 'W' wird der letzte Eintrag wiederholt, das heißt, beim Verkauf von 10 gleichen Artikeln braucht nicht 10 mal alle Information eingegeben zu werden, sondern nur einmal und anschließend wird 9 mal die Taste 'W' gedrückt.

Mit 'E' wird diese Eingabe beendet. Übrigens brauchen die Verkäufe nicht alle auf einmal eingegeben zu werden. Wenn bereits ein Eintrag in der Liste ist und der Menüpunkt wieder angesprochen wird, können die Daten in der Liste belassen werden. Der Inhalt dieser Liste ist mit der Option 2 aus dem Untermenü 'Tagesgeschäft' jederzeit ablesbar. Die Liste wird erst dann gelöscht, wenn die Tagesabrechnung durchgeführt ist und die verkauften Artikel aus dem Lagerbestand herausgezogen sind.

Diese Abrechnung (Punkt 3) wird zunächst durchgeführt, ohne den Lagerbestand zu verändern, erst wenn dies erwünscht ist, wird dieser Vorgang ausgeführt. So kann man auf Wunsch eine Abrechnung zwischendurch machen, um sich einen Überblick zu verschaffen.

Sicher sind die Programme nicht optimal auf die Bedürfnisse aller Kleinbetriebe abgestimmt. Sie lassen sich aber relativ leicht verändern und erweitern. Jedenfalls können sie ein Gerüst darstellen, an dem weiter gebaut werden kann.

Ohne Frage kann der Spectrum zu erheblichen Arbeitserleichterungen in einem kleineren Betrieb führen und es spricht nichts dagegen, ihn für solche Aufgaben auch einzusetzen.

ANHANG 1: DER BEFEHLSSATZ DES SPECTRUMS

Alle Anweisungen mit Ausnahme von INPUT, DEF und DATA können sowohl in einem Programm als auch als direkte Befehle eingegeben werden.Mehrere Anweisungen können in einer Zeile in einem Programm stehen, wenn sie durch einen Doppelpunkt getrennt sind. Zu beachten ist, daß vom ersten Auftreten einer REM Anweisung der Rest der Zeile ignoriert wird. Bei einer IF Anweisung werden alle folgenden Anweisungen in dieser Zeile nur ausgeführt, wenn die Anweisung nach THEN ausgeführt wird.

Hier nun eine vollständige Liste der Befehle für den Spectrum:

BEEP x,y	Für x Sekunden ertönt durch den eingebauten Lautsprecher ein Ton, dessen Höhe y Halbtöne über dem mittleren C liegt. Wenn y negativ ist, ist die Tonhöhe tiefer als C. Das Signal liegt auch am EAR Ausgang und kann verstärkt werden
BORDER m	Die Umrandung des Bildschirms sowie die Hintergrundfarbe des unteren Teils des Schirms erhalten die Farbe m. m liegt zwischen 0 und 7
BRIGHT n	Der Kontrast für die folgenden Zeichen wird festgelegt. n darf nur 0 (für normal), 1 (für scharf) und 8 (für transparent) sein
CAT	Gilt nur für Microdrives (siehe unten)
CIRCLE x,y,z	Zeichnet einen Kreis mit dem Mittelpunkt bei x,y und dem Radius z

CLEAR	Löscht alle Variable und schafft somit Speicherplatz. Weiterhin wird ein RESTORE und CLS durchgeführt, die PLOT Position in die linke untere Ecke (0,0) gesetzt und der GO SUB Speicher gelöscht
CLEAR n	Wie CLEAR, jedoch wird zusätzlich, falls möglich, RAMTOP auf n gesetzt
CLOSE #	Nur beim Betrieb der Microdrives gültig (siehe unten)
CLS	Löscht den Bildschirmspeicher und damit den Bildschirm
CONT(inue)	Setzt das Programm fort, und zwar an der Stelle, an der es mit einer anderen Meldung als mit 0 unterbrochen wurde
COPY	Kopiert die oberen 22 Zeilen des Bildschirms auf den Drucker, falls dieser angeschlossen ist
DATA a,b,c,...	Datenliste in einem Programm, die mit der READ Anweisung gelesen werden kann
DEF FN f(a,b,...)=e	Definiert eine Funktion, die mit FN im Programm aufgerufen werden kann
DELETE f	Gilt nur für Microdrive - Betrieb (siehe unten)
DIM a(m,n,...,s)	Löscht ein Feld a, falls vorhanden, und reserviert Speicherplatz für ein numerisches Feld a mit s Dimensionen. Alle Anfangswerte werden zu Null gesetzt

DIM a$(m,n,...,s)	Wie DIM a(m,n...,s), hier jedoch für ein Feld aus Strings. Alle Werte sind zu Beginn auf " " gesetzt
DRAW x,y	Zeichnet eine Gerade von der derzeitigen PLOT Position a,b zur Position a+x, b+y
DRAW x,y,z	Zeichnet einen Kreisausschnitt, der die derzeitige Plotposition a,b mit dem Punkt a+x, b+y verbindet. Dabei wird ein Anfangswinkel z im Bogenmaß genommen
ERASE	Nur für den Microdrive Betrieb gültig (siehe unten)
FLASH n	Stellt ein Blinken der Zeichen ein, falls n = 1, stellt es wieder ab, falls n = 0. Für n = 8 bleibt der alte Status erhalten
FOR a=x TO y	Löscht den Wert der Variablen a und setzt a als Kontrollvariable für eine Schleife auf. a nimmt zunächst den Wert INT x an und durchläuft die Schleife, deren Ende durch NEXT a gekennzeichnet sein muß. Danach wird der Wert von a um 1 erhöht und die Schleife erneut durchlaufen. Dieser Vorgang wird wiederholt, bis a > y erreicht wird
FOR a=x TO y STEP z	Wie FOR a=x TO y, jedoch wird hier die Kontrollvariable nach jedem Durchlaufen der Schleife um a+z geändert. Dabei kann z auch negativ sein (wenn x > y gilt)

FORMAT f	Nur für Microdrive Betrieb gültig (siehe unten)
GOSUB n	Bewirkt den Sprung zu einem Unterprogramm, welches an der Zeilennummer n beginnt. Das Unterprogramm muß mit einer RETURN Anweisung beendet werden
GO TO n	Bewirkt den Sprung zur Zeilennummer n. Falls diese Zeilennummer nicht vorhanden ist, wird die nächst größere Zeilennummer gewählt
IF x THEN y	Falls x, was auch ein Ausdruck sein kann, wahr ist, d. h. nicht Null, so wird die Anweisung y ausgeführt
INK n	Setzt die Farbe, in der die Zeichen gedruckt werden. Für n zwischen 0 und 7 wird die entsprechende Farbe gewählt, n = 8 bedeutet transparent, n = 9 steht für Kontrast
INPUT ...	Unterbricht das Programm und erwartet eine Eingabe, die mit ENTER abgeschlossen werden muß
INVERSE n	Tauscht Vorder- und Hintergrundfarbe aus falls n = 1. Mit n = 0 wird wieder der Normalmodus erreicht. Andere Werte für n sind nicht erlaubt
LET v = e	Ordnet der Variablen v den Wert e oder den Wert eines Ausdrucks e zu
LIST	Listet das Programm auf dem Bildschirm

LIST n	Listet das Programm von der ersten Zeilennummer, die mindestens n ist, auf den Bildschirm
LLIST	Gibt das Programmlisting auf den Drucker aus
LLIST n	Gibt das Programmlisting von der ersten Zeilennummer, die mindestens n ist, auf den Drucker aus
LOAD f	Lädt ein Programm mit allen Variablen
LOAD f DATA ()	Lädt ein numerisches Feld
LOAD f DATA $()	Lädt ein Stringfeld
LOAD f CODE m,n	Lädt n Bytes an die Speicheradressen m bis m + n
LOAD f CODE m	Lädt Bytes beginned an der Speicheradresse m
LOAD f CODE	Lädt Bytes an die Speicheradressen, von denen sie geSAVEd wurden
LOAD f SCREEN$	Identisch mit LOAD f CODE 16384, 6912. Lädt 6912 Bytes in den Bildschirmspeicher
LPRINT e	Ausgabe der Größe oder des Ausdrucks e auf den Drucker
MERGE f	Wie LOAD f, jedoch werden hier nicht die im Spectrum vorhandenen Programmzeilen gelöscht, es sei denn, sie sind identisch mit den hinzugeladenen

MOVE a,b	Nur für Microdrive Betrieb gültig (siehe unten)
NEW	Löscht das Basic Programm mit allen Variablen. Beeinflußt nicht die UDG oder den nach CLEAR n gesicherten Speicherplatz mit Adressen > n
NEXT x	Abschluss einer mit FOR begonnenen Schleife
OPEN #	Nur für Microdrive Betrieb gültig (siehe unten)
OUT m,n	Gibt das Byte n am Ausgang m aus
OVER n	Für n = 1 wird ein Zeichen auf dem Bildschirm geschrieben, ohne daß das vorherige Zeichen gelöscht wird. Mit n = 0 wird wieder der Normalmodus hergestellt
PAPER n	Wie INK, jedoch wird hier die Hintergrundfarbe gesteuert
PAUSE n	Hält den Programmablauf an und gibt den Bildschirmspeicher n Einheiten lang aus, wobei n = 1/50 Sekunde ist. Falls inzwischen eine Taste gedrückt wird, wird die Pause abgebrochen. Falls n = 0, wird die Pause solange durchgeführt, bis eine Taste gedrückt ist
PLOT m,n	Setzt einen Grafikpunkt an die Koordinaten m,n. Dabei gilt 0 <= m <= 255 und 0 <= n <= 175

POKE m,n	Schreibt den Wert n in den Speicherplatz m
PRINT e	Gibt den Wert oder Ausdruck e auf den Bildschirm an der nächsten PRINT Position aus
PRINT AT x,y;e	Gibt e an der Zeile x und der Spalte y aus. Dabei gilt 0<= x <= 21 und 0<= y <= 31
PRINT TAB x;e	Gibt e in der Spalte x der nächsten verfügbaren Zeile aus.
RAND(omize) n	Setzt den Anfangswert für den Zufallsgenerator auf den Wert n, falls n <> 0 ist. Ist n = 0, oder wird n weggelassen, wird für n die bisher auf dem Bildschirm gezeigte Anzahl der Bilder gewählt
READ a,b,c,...	Liest Werte aus einer DATA Anweisung und ornet diese den Variablen a, b, usw zu
REM	Abkürzung für Remark (Kommentar). Alle Zeichen in einer Programmzeile begonnen mit der REM Anweisung werden ignoriert
RESTORE n	Setzt den Zeiger für eine READ Anweisung auf die Zeile n. Wird n weggelassen, heißt dies RESTORE 0
RETURN	Ist die letzte Anweisung in einem Unterprogramm. Nach dieser Anweisung springt das Programm zu der Anweisung nach dem entsprechenden GOSUB

RUN n	Löscht alle Variablen und startet das Programm an der Zeile n. Wird n weggelassen, startet das Programm mit der ersten Zeile
SAVE f	Sichert das Programm zusammen mit allen Variablen
SAVE f LINE n	Sichert das Programm zusammen mit allen Variablen. Wird ein so gesichertes Programm geladen, wird ein automatischer Sprung zur Zeile n ausgeführt
SAVE f DATA ()	Sichert ein numerisches Feld
SAVE f DATA $()	Sichert ein Stringfeld
SAVE f CODE m,n	Sichert n Bytes von der Speicheradresse m an
SAVE f SCREEN$	Sichert den Bildschirmspeicher. Diese Anweisung ist identisch mit SAVE f CODE 16384,6912
STOP	Unterbricht den Programmablauf mit der Mitteilung 9. Wird hiernach CONT eingegeben, wird das Programm mit der auf STOP folgenden Anweisung fortgesetzt
VERIFY	Wie LOAD, jedoch werden in diesem Fall die Daten nicht in den Speicher geladen, sondern mit dem Inhalt des Speichers verglichen

Neben den Anweisungen versteht der Spectrum eine Reihe von Funktionen, die im folgenden gelistet sind:

Funktion	Bedeutung
ABs x	Absoluter Wert von x
ACS x	Arcuscosinus von x in Radians. Es muß gelten -1 <= x <= 1
AND	Logische Verknüpfung. x AND y ist wahr, falls x = y
ASN x	Arcussinus von x in Radians. Es muß gelten -1 <= x <= 1
ATN x	Arcustangens von x in Radians. Es muß gelten -1 <= x <= 1p
ATTR (x,y)	Eine Zahl, die in binärer Form die Attribute der Position des Bildschirms mit der Zeilenzahl x und der Spaltenzahl y angibt. Bit 7, das Signifikanteste, ist 1 für Flash 1 und 0 für Flash 0. Bit 6 ist 1 für Bright, 0 für normal, Bit 5 bis 3 geben die Hintergrundfarbe und BIT 2 bis 0 die Vordergrundfarbe an
BIN	Möchte man eine Zahl nicht in dezimaler sondern in binärer Form eingeben, so kann dies durchgeführt werden, indem zunächst BIN gefolgt von Nullen und Einsen eingegeben wird
CHR$ x	Ergibt das Zeichen, dessen Code INT x ist
CODE x$	Gibt den Code des ersten Zeichens des Strings x$. Handelt es sich um einen Leerstring, ist das Ergebnis 0
COS x	Cosinus x
EXP x	Exponentialfunktion von x, also e hoch x

FN e(x) Hiermit wird eine durch DEF FN definierte Funktion aufgerufen. Das Argument x muß immer in Klammern stehen. Gibt es kein Argument, müssen die Klammern dennoch eingegeben werden

IN x Gibt den Wert, der am Eingang x eingegeben wurde

INKEY$ Überprüft die Tastatur auf eine gedrückte Taste. Das Ergebnis ist das Zeichen, welches durch die entsprechende Taste dargestellt wird. Ist keine Taste gedrückt, so ist das Ergebnis "", d. h. ein Leerstring.

INT x Gibt den ganzzahligen Teil der Zahl x. x - INT x gibt den Teil von x hinter dem Komma

LEN x$ Gibt die Länge des Strings x$ an

LN x Ergibt den natürlichen Logarithmus der Zahl x

NOT x Logische Verknüpfung. Der Ausdruck ist 0, wenn x <> 0 ist und 1, falls x = 0 gilt

OR Logische Verknüpfung. x OR y bedeutet, mindestens x oder y muß wahr sein

PEEK x Gibt den Wert des Bytes im Speicherplatz INT(x) an

PI Steht für die Konstante 3.14159265....

POINT (x,y) Ergibt 1, falls für den Punkt x,y eine Vordergrundfarbe gesetzt wurde, und 0, falls dieser Punkt eine Hintergrundfarbe darstellt. Es muß gelten 0 <= x <= 255 und 0 <= y <= 175

RND Gibt eine Pseudo - Zufallszahl zwischen 0 und 1. Eine Zahl zwischen a und b wird durch die

Anweisung a + (b-a)*RND erzeugt

SCEEN$(x,y)	Ergibt das in Zeile x und Spalte y stehende Zeichen. Falls das Zeichen nicht erkannt wird, wird ein Leerstring ausgegeben
SGN x	Ergibt das Vorzeichen der Variablen x. Das Ergebnis ist -1, falls x<0, 0, falls x = 0 und 1, falls x > 0 ist
SQR x	Gibt die Quadratwurzel aus x
STR$ x	Die Zahl x wird in einen String umgewandelt
TAN x	Tangens x in Radians
USR x	Ruft eine in Maschinensprache geschriebene Routine auf, deren Anfangsadresse am Speicherplatz x ist
USR x$	Gibt die Adresse für das Bit Muster eines selbstdefinierbaren Zeichens
VAL x$	Gibt den numerischen Wert des Strings x$
VAL$ x$	Gibt den numerischen Wert des Strings x$ wieder als String
- x	Führt einen Vorzeichenwechsel der Variablen x durch

Folgende binären Operationen kann der Spectrum durchführen:

+	Addition (auch von Strings)
-	Subtraktion
*	Multiplikation

/	Division
^	Potenzieren
=	Gleichsetzen
>	grösser als
<	kleiner als
>=	grösser oder gleich
<=	kleiner oder gleich
<>	ungleich

Mit dem Interface 1 wird der Befehlssatz des Spectrums erweitert. Es gibt nun 15 Streams, die mit Zahlen von 1 bis 15 gekennzeichnet werden. Weiterhin existieren sieben Arten von Kanälen (Channels), die mit "k" für Keyboard, "s" für Screen, "p" für Printer, "t" für Textinterface für die RS232 Schnittstelle, "b" für Binärinterface für die RS232 Schnittstelle, "n" für das Netzwerk und "m" für die Microdrives bezeichnet werden. Die folgenden Anweisungen sind nun möglich:

CAT x	Gibt ein Inhaltsverzeichnis der Microdrive - Kassette im Microdrive x. Die Liste wird alphabetisch ausgegeben. Weiterhin wird der Name der Kassette und der freie Speicherplatz in KBytes angegeben
CAT #z;x	Gibt das Inhaltsverzeichnis der Microdrive Kassette in Microdrive x über den Strom z aus
CLOSE # x	Entkoppelt alle Kanäle vom Strom x. Falls noch Daten im Zwischenspeicher (buffer) vorhanden sind, werden diese ausgegeben
ERASE"m";x;f	Löscht den File mit Namen f von der Microdrive - Kassette im Microdrive x
FORMAT"m";x;f	Formatiert eine Microdrive - Kassette im Microdrive x. Dabei wird der Kassette der Name f gegeben, der bei jedem Inhaltsverzeichnis mit ausgegeben wird. Ist die Kassette bereits beschrieben gewesen, so sind alle Daten gelöscht
FORMAT"n";x	Gibt dem Spectrum die Nummer x im Rahmen des Netzwerkes

FORMAT"t";x	Setzt die Übertragungsrate von Daten über die RS232 Schnittstelle auf x. x wird als einer der Werte 50, 110, 300, 600, 1200, 2400, 4800, 9600 oder 19200 angenommen.
FORMAT"b";x	Wie FORMAT"t";x. Für t- und b- Kanal ist die Baudrate immer gleich
INKEY #x	Gibt das Zeichen wieder, welches vom Strom x kommt. Liegt kein Zeichen vor, so wird der Leerstring zurückgegeben, also "".
INPUT #x;y	Eine Größe y wird vom Strom y aus eingegeben. Die LINE Option ist auch hier möglich
LOAD*...	Lädt ein Programm, Daten oder Bytes aus dem spezifizierten Kanal. Nur die Kanäle "b", "n" und "m" sind erlaubt. Alle Optionen, die vom 'normalen' LOAD bekannt sind, sind erlaubt
MERGE*...	Wie das gewöhnliche MERGE, nur vom spezifizierten Kanal aus
MOVE a TO b	Überträgt Daten von der Quelle a zum Ziel b. Es ist nicht notwendig, vor dieser Anweisung einen Kanal zu öffnen oder nachher zu schliessen
OPEN #x,y	Der Kanal y wird dem Strom x zugeordnet. Dadurch kann in Basic Eingabe und Ausgabe an diesen Kanal durchgeführt werden. Erlaubte Ströme sind "k", "s" und "p"
PRINT #x....	Führt eine Ausgabe zum spezifizierten Strom aus. Dieser muß vorher als Aus-

gabestrom geöffnet worden sein

SAVE * ...	Sichert ein Programm, Daten oder Bytes auf dem spezifizierten Kanal. Nur die Kanäle "b", "n" und "b" dürfen benutzt werden. Alle Optionen des 'normalen' SAVE sind vorhanden
VERIFY * ...	Wie das 'normale' VERIFY, nur kommen die Daten vom spezifizierten Kanal

ANHANG 2: FEHLERMELDUNGEN

Jedesmal, wenn der Spectrum einen Befehl oder ein Programm ausgeführt hat, gibt er eine Meldung am unteren Bildschirmrand aus. Diese Meldung besteht aus einem Code und einer kurzen Erklärung, welche gegebenenfalls die Fehlersuche erleichtert.

In der Regel ist die wünschenswerteste Meldung '0 OK, Z(eile):B(efehl)'. Diese besagt, daß die Operation ordnungsgemäß erledigt wurde und kein formeller Fehler aufgetreten ist. Leider werden aber häufig auch andere Mitteilungen erscheinen. Die folgende Liste gibt alle Meldungen wieder.

0 OK
Erfolgreicher Ablauf des Programms

1 NEXT without FOR
In diesem Fall existiert zwar das Ende einer Schleife, der Anfang, also die FOR Anweisung, wurde vergessen

2 Variable not found
Bei einer einfachen Variablen tritt dieser Fehler auf, wenn die Variable benutzt wird, ehe sie mit LET, READ oder INPUT definiert wurde. Bei einem Feld tritt dieser Fehler auf, wenn die DIM Anweisung vergessen wurde

3 Subscript wrong
Eine Feldvariable hat einen größeren Wert, als in einer LET oder DIM Anweisung festgelegt

4 Out of memory
Der Speicherplatz des Spectrums reicht nicht aus

5 Out of screen
Bei einer INPUT Anweisung wurden mehr als 23 Zeilen

in der unteren Bildschirmhälfte erzeugt, oder eine PRINT Anweisung verlangt eine Zeilennummer größer als 21

6 Number too big
In einer Rechnung tritt eine Zahl größer 10 hoch 38 auf. Meist liegt dies an einer Division durch eine Zahl, die den Wert 0 zugeordnet hat

7 RETURN without GO SUB
Das Programm läuft auf ein RETURN, ohne vorher eine GO SUB Anweisung erhalten zu haben

8 End of file
Ein Ende eines Files wurde gefunden. Dies hat nur für Microdrive Benutzer Bedeutung

9 STOP Statement
Das Programm ist durch eine STOP Anweisung unterbrochen worden. Gibt man nun CONTINUE ein, so wird eine Anweisung hinter der STOP Anweisung weiter gemacht

A Invalid argument
Das Argument einer mathematischen Funktion ist nicht legal

B Integer out of range
Eine ganzzahlige Zahl ist zu groß geraten

C Nonsense in BASIC
Ein nicht sehr beliebter Kommentar bei Basic-Programmierern. Der durch VAL berechnete Ausdruck ist nicht legal

D BREAK - CONT repeats
Das Programm wurde durch Betätigen der BREAK Taste unterbrochen. Gibt man anschliessend CONT ein, so wird die unterbrochene Anweisung wiederholt, im Gegensatz zum Verfahren bei der Meldung L (siehe weiter unten)

E Out of DATA

Es sollen mehr Größen mit einer READ Anweisung gelesen werden, als mit DATA eingegeben wurden. Oft wurde lediglich ein RESTORE vergessen

F Invalid file name

Es wurde versucht, einen File entweder ohne Namen oder mit einem Namen, der mehr als 10 Zeichen umfaßt, zu SAVEn

G No room for line

Es ist nicht mehr genug Speicherplatz für die nächste Programmzeile vorhanden

H STOP in INPUT

Eine INPUT Anweisung wurde mit der STOP Taste beantwortet. CONTinue wiederholt die INPUT Anweisung

I FOR without NEXT

Der Anfang einer Schleife wurde durch die FOR Anweisung gemacht, jedoch vermißt der Spectrum den Abschluß durch NEXT

J Invalid I/O device

Fehlerhafte Input oder Output Anweisung bei Microdrives

K Invalid colour

Die für eine Farbe eingegebene Zahl ist nicht legal

L BREAK into programm

Zwischen der Ausführung zweier Programmzeilen wurde das Programm durch Drücken der BREAK Taste unterbrochen. In diesem Fall bewirkt die anschliessende Eingabe von CONTinue die Ausführung des nächsten Befehls

M RAMTOP no good

Die angegebene Zahl für RAMTOP ist entweder zu groß oder zu klein gewählt

N	Statement lost
Es liegt ein Sprung auf eine Anweisung vor, die nicht mehr existiert

O	Invalid stream
Bei Benutzung von Microdrives bzw Interface 1 wurde eine illegale Stream Anweisung gegeben

P	FN without DEF
Eine Funktion wurde benutzt, ohne sie vorher durch eine DEF FN zu definieren

Q	Parameter error
Bei einer mit DEF FN definierten Funktion wurde bei einem Aufruf eine falsche Anzahl oder ein falscher Typ von Parametern angegeben

R	Tape loading error
Ein Lesefehler beim Laden oder Verifizieren vom Band wurde gefunden

Wenn Sie ein Interface 1 an den Spectrum angeschlossen haben, können Sie in den Genuß von weiteren 21 Fehlermeldungen kommen:

- Code error
  Es wurde versucht, mit einer LOAD Anweisung mehr Code zu laden, als Platz vorhanden ist

- Drive 'write' protected
  Der Versuch, eine schreibgeschützte Microdrive - Kassette zu beschreiben, ist fehlgeschlagen

- File not found
  Entweder der durch eine LOAD Anweisung zu ladene File existiert nicht auf der Kassette oder er kann nicht vollständig eingelesen werden. Der letztere Fall tritt ein, wenn die Kassette teilweise beschädigt ist oder wenn der File nicht ordnungsgemäß geschlossen wurde

- Invalid device expression
  Eine Einheit wurde mit einem anderen als den legalen Buchstaben s,p,k,m,n,t,oder b spezifiziert. Dieser Fehler kann auch auftreten, wenn bei der Spezifikation des s,p oder k Kanals ein Semikolon anstelle eines Kommas verwendet wurde

- Invalid name
  Einem File wurde entweder kein Name oder ein Name mit mehr als 10 Zeichen zugeordnet

- Invalid station number
  Einer Netzwerkstation wurde eine Nummer kleiner 0 oder größer 64 zugeordnet oder es wurde bei der FORMAT Anweisung eine Zahl kleiner 1 oder größer 64 verwendet

- Invalid stream number
  Einem Strom wurde eine Zahl kleiner 0 oder größer 15 zugeordnet

- Merge error
  Es wurde versucht, Daten oder Code zu MERGEn. MERGE ist nur für Programme erlaubt und hier auch nur für solche, die nicht mit einer LINE Anweisung geSAVEd wurden

- Microdrive full
  Es ist kein Platz mehr auf der Microdrive - Kassette vorhanden, um die SAVE Anweisung auszuführen. Ein eventuell geöffneter File sollte von dieser Kassette ERASEd werden

- Microdrive not present
  Es wurde entweder ein Microdrive angesprochen, der eine unFORMATierte Kassette enthält, gar keine Kassette enthält oder der selbst nicht angeschlossen ist

- Missing baud rate
  Es wurde vergessen, die Baudrate mit der FORMAT Anweisung festzulegen

- Missing drive number
  Beim Versuch, auf einen Microdrive zuzugreifen, wurde vergessen, die Microdrivenummer anzugeben

- Missing name
  Ein Filename wurde nicht angegeben

- Missing station number
  Eine Stationsnummer wurde nicht angegeben

- Programm finished
  Es wurde versucht, eine Zeilennummer auszuführen, die größer als die letzte im Programm enthaltene ist. Diese Meldung tritt auch auf, wenn der RUN Befehl eingegeben wurde, ohne daß ein Programm im Speicher ist

- Reading a 'write' file
  Der Versuch wurde unternommen, Daten von einem nicht existierenden File zu lesen, oder von einem File, der zum Schreiben geöffnet wurde

- Stream already open
  Es wurde versucht, einen Stream zu öffnen, der bereits zu einem neuen Kanal zugeordnet ist(m,n,t oder b). In diesem Fall muß der Stream zunächst geschlossen werden

- Verification has failed
  Es wurde ein Unterschied zwischen einem geSAVEten File und dem entsprechenden File im Speicher während der Verifikation entdeckt

- Writing to a 'read' file
  Es wurde versucht, einen bereits existierenden File erneut auszugeben, ohne den alten File vorher zu ERASEn

- Wrong file type
  Für die gewünschte Operation ist der Filetyp nicht geeignet. Dies tritt zum Beispiel auf, wenn Code als Programm geladen werden soll oder ein Programm geMOVEd werden soll



Die Neuen Bücher von DATA BECKER



Wer sich für Elektronik interessiert und mehr aus seinem Sinclair Spectrum machen möchte, der findet hier das ideale Buch. Beschreibung der Hardwaregrundlagen – Parallel-In-Out-Interface – Centronics Druckerschnittstelle – Sound Box – A/D Wandler – Erweiterungskarte mit 5 Steckplätzen – EPROMMER – Sprachausgabe – Speicheraufrüstung bis 80 KB und vieles mehr. Dazu Anschlußbilder der wichtigsten IC's, Platinenlayouts und Bestückungspläne.

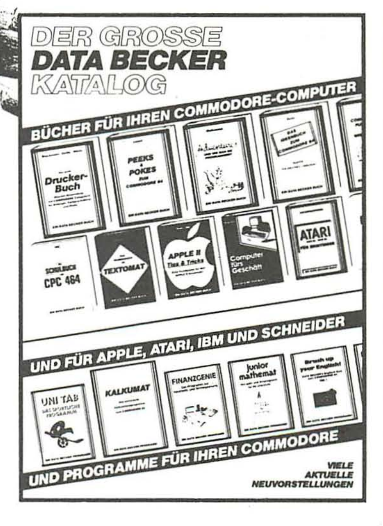
SINCLAIR SPECTRUM HARDWARE-ERWEITERUNGEN, ca. 320 Seiten, DM 49,–



Sie wollen mehr aus Ihrem Computer machen?

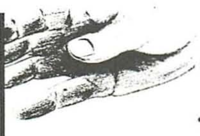
Da steht alles drin:

reingucken und durchblicken



Ihr Computer ist nichts wert ohne entsprechende Bücher und Programme. Und die finden Sie in diesem Katalog.
Über (30!) Superbücher zum COMMODORE 64, aber auch Bücher für APPLE, ATARI, IBM und SCHNEIDER Anwender. Dazu wichtige Software-Trainigsbücher und aktuelle Einsteigerliteratur.
Spitzenprogramme besonders für den COMMODORE 64, von der Textverarbeitung über die verschiedensten Programmierhilfen bis hin zur intelligenten Datenbank. Für Einsteiger, Profis, Freaks und Geschäftsleute.
Den großen DATA BECKER Katalog gibt's ab Ende Oktober bei Ihrem Fachhändler, natürlich kostenlos, oder direkt von DATA BECKER.





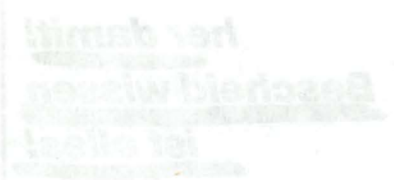
her damit! Bescheid wissen ist alles!

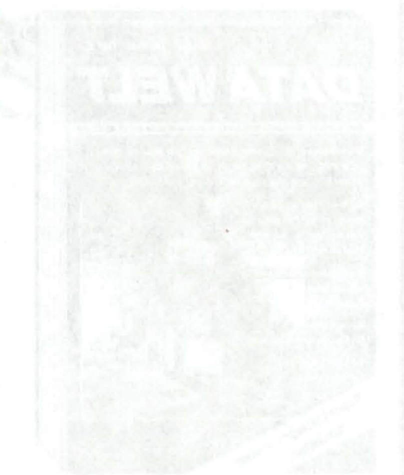
Die neue DATA WELT, das aktuelle Computermagazin aus dem Hause DATA BECKER. Ein starkes Blatt mit 140 Seiten, die es in sich haben. Randvoll mit aktuellen Informationen, Tips und Tricks. Mit aktuellen Hintergrundberichten. Mit der großen Marktübersicht.
Mit aktuellen Softwarepremieren. Mit Tips und Tricks zu DATAMAT und TEXTOMAT.
Mit interessanten Programmlistings auf über 30(!) Seiten. Und mit vielem Anderen mehr.

Die neue DATA WELT – jetzt am Kiosk und überall, wo es DATA BECKER Bücher und Programme gibt.

DATA BECKER

Merowingerstr. 30 · 4000 Düsseldorf · Tel. (0211) 310010 · im Hause AUTO BECKER







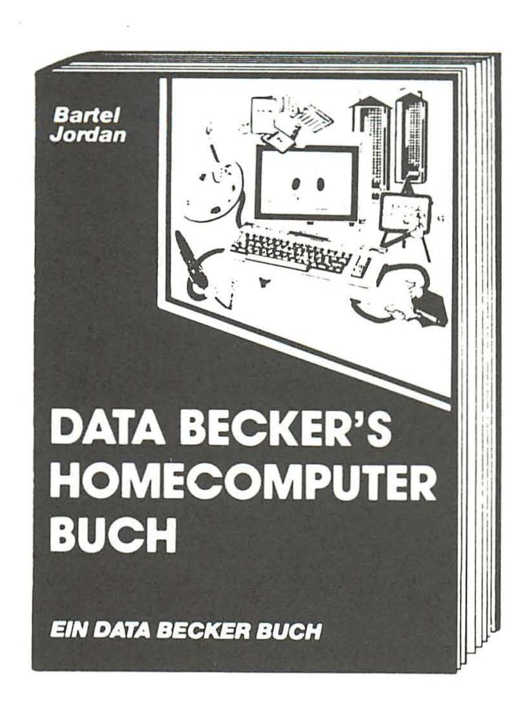
Die Neuen Bücher von DATA BECKER



Auf dieses Buch haben Manager, Unternehmer, Freiberufler und all diejenigen gewartet, die sich für den beruflichen und geschäftlichen Einsatz eines Mikrocomputers interessieren. Leicht verständlich, kompetent und ohne jedes „Computer-Chinesisch“ zeigt es, was ein Computer für Sie tun kann. Um das Thema Computer kommen Sie nicht mehr herum. Dieses Buch hilft Ihnen dabei.

COMPUTER FÜRS GESCHÄFT, ca. 250 Seiten, DM 39,–





Faszinierend, was so ein Homecomputer alles kann. Dieses leicht verständliche Buch, das keinerlei Computerkenntnisse voraussetzt, hilft Ihnen nicht nur bei der richtigen Kaufentscheidung. Es berät Sie auch umfassend beim sinnvollen Einsatz Ihres eigenen Computers. Wichtige Informationen, wertvolle Ideen und nützliche Vorschläge zum Thema **HOMECOMPUTER auf über** 380 **Seiten für nur DM 29,–**

Die Neuen Bücher von DATA BECKER



DAS STEHT DRIN:

ZX-Spectrum-Tips & Tricks enthält eine hochinteressante Sammlung von Anregungen, Ideen und fertigen Lösungen zur Programmierung und Anwendung Ihres ZX-Spectrum.

Aus dem Inhalt:

- 64 Zeichen pro Zeile
- Absturzsichere Eingaben
- Symbolraten (Spiel)
- Grafische Darstellung einer Weltkarte
- Anschluß und Nutzungsmöglichkeiten von Microdrives bis Lightpen
- Programme für Säulen- und Kreisdiagramme
- Lager- und Umsatzverwaltung
- Kundendatei
- Programm zur Schaufensterwerbung
- und natürlich mit vielen PEEKs, POKEs und USRs, um ROM und RAM optimal zu nutzen.

UND GESCHRIEBEN HAT DIESES BUCH:

Dr. Peter Bosetti ist Physiker an der Universität Aachen. Die Wartezeiten am Großrechner brachten ihn zum Sinclair ZX-Spectrum, der ihn so faszinierte, daß er seine Erfahrungen, Ideen und Tricks in diesem Buch veröffentlichte.

ISBN 3-89011-075-4